A SOLUTION-ADAPTIVE ALGEBRAIC GRID GENERATION METHOD BASED ON VARIATIONAL PRINCIPLES

BY

JULIO SERGIO DOLCE DA SILVA

A DISSERTATION PRESENTED TO THE GRADUATE SCHOOL
OF THE UNIVERSITY OF FLORIDA IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF DOCTOR OF PHILOSOPHY

UNIVERSITY OF FLORIDA

1988

To my parents

Geraldo Siqueira da Silva

and

Clarice Dolce da Silva.

ACKNOWLEDGMENTS

I would like to express my thanks to Dr. Vernon P. Roan for being the chairman of my dissertation committee, for his guidance, and for his support since the beginning of my studies. I also would like to thank Dr. Tom I-P. Shih, my cochairman, for suggesting this study, for his continuous orientation, and for his proper advices during the course of this work.

Thanks are also due to Dr. R. B. Gaither, chairman of the Mechanical Engineering Department and member of my committee, for his encouragement and support. I am also indebted to Dr. J. Hammack, Jr., and Dr. H. A. Ingley for serving as members of my dissertation committee.

Finally, I would like to extend my acknowledgments to my wife, Marianna, to my daughter, Clarissa, and to my son, George, for their confidence, encouragement, and patience.

TABLE OF CONTENTS

KEY TO SYMBOLS

$\vec{a}$	Vector defined by two grid points
A_1	Partial derivative of the coordinate x_1 with respect to ξ
A_2	Partial derivative of the coordinate x_2 with respect to ξ
[A]	Matrix of the metric coefficients
Ag	Cell area of the grid system
Ag_{crt}	Minimum cell area allowed
B_1	Partial derivative of the coordinate y_1 with respect to ξ
B_2	Partial derivative of the coordinate y_2 with respect to ξ
c	Constant of Burgers' equation (m/sec)
C_1, C_2	Constants of integration
D	y-coordinate of the vertex of a parabola
D_i	Distance between the boundaries
D_m	Minimum diameter of a convergent-divergent nozzle
$\vec{e}_\xi$	Normal vector to the boundary
$\vec{e}_\eta$	Tangent vector to the boundary
FDEs	Finite difference equations
G	Property of the grid system related to the grid spacing
$\vec{i}$	Unitary vector in x-coordinate direction
IL	Number of grid points in the x-coordinate direction
$\vec{j}$	Unitary vector in y-coordinate direction
J	Jacobian of a coordinate transformation
J_{crt}	Minimum Jacobian allowed

Jmax	Maximum Jacobian
Jmin	Minimum Jacobian
JL	Number of grid points in the y-coordinate direction
JL^*	Number of grid points in the ξ-direction after stretching
k	Number of iterations
K1, K2	Coefficients of Hermite interpolation
Ka	Minimum coefficient for clustering
Kb	Maximum coefficient for clustering
Kc	Minimum coefficient for stretching
K(x)	Stretching function
L_1, L_2	Boundaries of the physical domain in x-direction
L_1', L_2'	Boundaries of the physical domain in y-direction
L_{ij}	x-coordinate of grid point ij at the boundaries
L_{ij}'	y-coordinate of grid point ij at the boundaries
n	Time level
P	Grid point in the interior of the physical domain
P_o	Grid point at the boundaries of the physical domain
P(x)	Parametric function for the y-coordinates and η=0.5
Q(x)	Parametric function for the x-coordinates and η=0.5
$\vec{r}$	Vector position of an interior grid point
SAAGG	Solution adaptive algebraic grid generation
SF	Smoothness factor
t	Time coordinate in the physical domain
T	Duration of interest
T_p	Computational time per grid point
T_t	Total computational time

u	Dependent variable
V	Vertex of a parabola
W	Weighting function
x, y	Coordinates of the physical domain before stretching
$\bar{x}, \bar{y}$	Coordinates of the physical domain after stretching
α	Ratio between Δ_m and Δ_b
β	Minimum angle of deviation from orthogonality
Δb	Distance between the grid points s and r (Fig. 5.3)
ΔL_j	Length of the j^{th} grid line in the physical domain
$\Delta L_i'$	Length of the i^{th} grid line in the physical domain
Δm	Distance between the grid points m and t (Fig. 5.3)
Δn^*	Grid spacing in the physical domain after stretching
ΔS_{ij}	Length of the j^{th} grid line from i-1 to i
Δt	Time increment
$\Delta x, \Delta y$	Grid spacing in the physical domain before stretching
$\Delta\bar{x}, \Delta\bar{y}$	Grid spacing in the physical domain after stretching
Δx_{crt}	Minimum grid spacing allowed in the physical domain
Δx_{max}	Maximum grid spacing in the physical domain
Δx_{min}	Minimum grid spacing in the physical domain
$\Delta\xi, \Delta\eta$	Grid spacing in the computational domain
ξ, η, τ	Coordinates of the computational domain
θ	Angle of deviation from orthogonality
ω	Weighting function before being raised to SF power

Abstract of Dissertation Presented to the Graduate School of the University of Florida in Partial Fulfillment of the Requirements for the Degree of Doctor of Philosophy

A SOLUTION-ADAPTIVE ALGEBRAIC GRID GENERATION METHOD BASED ON VARIATIONAL PRINCIPLES

By

Julio Sergio Dolce da Silva

April, 1988

Chairman: Dr. Vernon P. Roan
Cochairman: Dr. Tom I-P. Shih
Major Department: Mechanical Engineering

A solution-adaptive, algebraic grid generation method was developed for generating grid points that are needed by finite-difference algorithms to obtain numerical solutions to complex fluid flow problems. The grid generation technique developed is ideally suited for use in the analysis of unsteady, multidimensional fluid flow problems with deforming spatial domains and problems with disparate length scales in different parts of the spatial domain which can move about in an unpredicable manner.

The grid generation technique developed in this investigation is based on variational principles and does not require solution of partial differential equations to generate grid points. The method will automatically cluster grid points in regions where needed to resolve all significant length scales of the problem being analyzed. In addition,

this grid generation technique will ensure that the grid system generated is smooth and nearly orthogonal.

The orthogonality of the grid system is controlled by means of parameters introduced to ensure that the deviation from orthogonality will be within certain limits, fixed a priori.

To demonstrate the usefulness of the solution-adaptive, algebraic grid generation technique developed in this investigation, the method was used with finite-difference algorithms to analyze (1) the linearized one-dimensional inviscid Burgers' equation, (2) the quasi-linear, one-dimensional inviscid Burgers' equation, (3) the two-dimensional unsteady problem of plane wave propagation, and (4) the propagation of concentric, circular waves in the radial direction. Also, the method was used in conjunction with an algebraic grid generation method known as Two-Boundary Technique to generate a grid network in a convergent-divergent-shaped spatial domain.

Solutions were obtained for several problems and compared with known solutions. These comparisons indicated that the solution-adaptive, algebraic grid generation method developed in this investigation is useful and can, efficiently, generate grid points required by finite-difference algorithms to obtain numerical solutions for fluid flow problems.

CHAPTER I

INTRODUCTION

1.1 Background

Computational fluid dynamics has experienced tremendous advances during the past two decades and has emerged as one of the most powerful tools for analyzing practical fluid flow problems. One of the most important factors contributing to the success of computational fluid dynamics as a tool for analyzing fluid flow problems has been the advances made in grid generation techniques. The evolution of grid generation techniques has markedly expanded the scope of fluid flow problems that can be analyzed by numerical methods. Grid generation techniques supported these advances because they enabled computational fluid dynamics to analyze, efficiently and accurately, problems with very complex geometries, problems with deforming spatial domain as well as problems with highly disparate length scales in different parts of the domain. Grid generation techniques in conjunction with finite-difference methods constitute one of the most powerful tools for analyzing practical fluid flow problems [1].

Before the advent of grid generation techniques, complicated interpolation schemes were needed to implement boundary conditions when finite-difference methods were used to analyze fluid flow problems with irregularly shaped spatial domains. At that time, those numerical methods apparently were not able to solve fluid flow problems with

complex-shaped boundaries or problems with disparate length scales in different parts of spatial domain that can move about. The development of grid generation techniques removed those limitations, gave new impetus to numerical methods of analysis, and made finite-difference methods versatile and accurate in analyzing fluid dynamics problems with irregularly shaped and/or deforming spatial domains [2].

Grid generation is concerned with the mapping of optimally distributed grid points which may be non-equally spaced and/or moving in the physical domain onto a computational domain where all of the grid points are uniformly distributed and stationary. This invariably involves the generation of a coordinate system known as the boundary-fitted coordinate system for the computational domain. The boundary-fitted coordinate system is said to be generated once the coordinate transformation between the boundary-fitted coordinate system and the coordinate system of the physical domain is known. The boundary-fitted coordinate system is a curvilinear coordinate system in which one set of coordinate lines or surfaces, analytically, always coincides with the boundaries of the physical domain regardless of its shape or motion [3].

Once the boundary-fitted coordinate system has been established, the governing equations are transformed so that the coordinates of the boundary-fitted coordinate system become the independent variables instead of the coordinates of the physical domain. Afterwards, the finite-difference equations needed to obtain numerical solutions are derived from these transformed governing equations. The algebraic equations thus derived are analyzed in the computational domain where all of the grid points are stationary and uniformly distributed. Performing

calculations in this computational domain is important because it permits the development of general numerical codes that can be applied to solve many different fluid flow problems with different geometries by simply changing the boundary-fitted coordinate system and the boundary conditions.

At this point, it is important to note that boundary-fitted coordinate systems should satisfy several properties as described below.

First, the coordinate transformation between the boundary-fitted coordinate system and the coordinate system of the physical domain should be continuous and differentiable with a Jacobian that is never zero so that the mapping will be one to one.

Second, in order to facilitate implementation of boundary conditions, one set of grid points should always coincide with the boundaries of the physical domain regardless of its shape or motion. That is why one set of coordinate lines or surface of the boundary-fitted coordinate system should always coincide with the boundaries of the physical domain.

Third, for computational efficiency, the number of grid points used should be kept to the minimum required to resolve accurately all significant length scales of the problem. Thus, in order to achieve the same accuracy, grid points are often moving and nonuniformly distributed within the physical domain with more grid points in those regions where higher resolution is needed. The process of concentrating more grid points in certain regions is referred to as the stretching of the boundary-fitted coordinate system.

Fourth, when grid points in the physical domain are nonuniformly

distributed because of stretching, the boundary-fitted coordinate system should be smooth; i.e., the spacing between grid points should change gradually from a region where grid spacings are small to a region where grid spacings are large. Typically, the ratio of two adjacent grid spacings should be approximately between 0.5 and 2 [3].

Fifth, the boundary-fitted coordinate system should be such that the grid system in the physical domain is nearly orthogonal. This means that, the angle between intersecting grid lines, at the interior grid points should be, approximately, between 45 and 135 degrees. Mastin [4], showed that the factor which causes an increase in truncation error due to departure from orthogonality is directly proportional to the inverse of the sine of the angle between intersecting grid lines in the physical domain. Thus, some deviation from orthogonality has negligible effects on truncation errors and many examples were given in the reference cited to demonstrate this. However, grid lines should always be, approximately, orthogonal to the boundaries of the physical domain to facilitate implementation of derivative boundary conditions.

The ideal boundary-fitted coordinate system is one that possesses all of the five properties discussed above. In addition, the distribution of grid points in the physical domain should be coupled dynamically to the numerical solution as it is being computed so that grid points in the physical domain will automatically cluster in regions where they are needed and when they are needed. Such a grid system is referred to as a solution-adaptive grid system.

For many important fluid flow problems, the positions of those regions requiring a higher concentration of grid points are known a

priori and a satisfactory boundary-fitted coordinate system can be established by means of relatively simple techniques. However, for other problems in which those regions move about in an unpredictable manner, a solution-adaptive grid generation technique is needed. Solution-adaptive, grid generation techniques generally utilize variational principles to satisfy those requirements of stretching, smoothness, and orthogonality of the boundary-fitted coordinate system as it is being generated.

All solution-adaptive, grid generation methods developed so far can be classified into two major categories: (1) algebraic grid generation methods and (2) differential equation grid generation methods. Algebraic grid generation methods use interpolation techniques to generate boundary-fitted coordinate systems and can provide precise controls over the distribution of grid points in the physical domain by means of stretching functions. Also, they are highly efficient because they do not need to solve partial differential equations to generate boundary-fitted coordinate systems. Disadvantages of algebraic grid generation methods include difficulties in controlling smoothness and orthogonality.

Differential equation grid generation methods generate boundary-fitted coordinate systems by solving systems of partial differential equations. Since those systems of partial differential equations must be solved numerically, the differential equation grid generation methods are less efficient than algebraic grid generation methods. For three dimensional, solution-adaptive grid systems in which a different grid system must be generated at each time step, differential equation

grid generation methods may not be feasible, because, in this case they become computationally very expensive. Most solution-adaptive, grid generation methods developed so far utilize variational principles while constructing boundary-fitted coordinate systems. Variational principles have been employed to optimize those requirements of stretching, smoothness and orthogonality of the boundary-fitted coordinate system. Variational techniques utilized while generating boundary-fitted coordinate systems can be used with either differential equation grid generation methods or algebraic grid generation methods. The method developed in this investigation was a solution-adaptive, algebraic grid generation method and variational principles were utilized.

1.2 Literature Survey

Efforts have been made during this past decade to develop grid generation techniques. All grid generation techniques developed so far can be divided into two major categories: (1) non-adaptive, grid generation methods and (2) solution-adaptive grid generation methods. Non-adaptive grid generation methods are designed for steady problems and problems in which regions of sharp gradients are known a priori so that grid points can be clustered correctly to improve the accuracy of the numerical solution and at the same time reduce the number of grid points needed to solve the problem. Solution-adaptive grid generation methods are designed for unsteady fluid flow problems with deforming spatial domain and/or problems with disparate length scales in different parts of the spatial domain which can move about in an unpredictable manner. As noted earlier, solution-adaptive grid generation methods can

be further divided into the following categories: (1) algebraic grid generation methods and (2) differential equation grid generation methods.

An authoritative and comprehensive review of grid generation techniques was presented by Thompson, Zahir, Warsi and Mastin [5]. Their review covered, in depth, differential equation grid generation methods, algebraic grid generation methods, and solution-adaptive grid generation techniques, developed before 1980. Since that time many important investigations have been done in grid generation. This literature survey discusses the advances made in solution-adaptive grid generation methods since 1980.

Hindman and Spencer [6] developed a solution-adaptive, grid generation scheme based on elliptic partial differential equations with a time-dependent control function. The time-dependent control function dynamically couples the numerical solution and the speed at which each grid point should travel to solve the physics of the problem being investigated. A new boundary-fitted coordinate system is generated at each time step based on the computed grid speeds. This method suffers from cumulative errors in the computed grid speeds so that grid point locations, after some time interval, need to be readjusted by using a non-adaptive, grid generation technique. Although their method is general and can be applied to analyze multidimensional problems, it only has been applied to analyze one-dimensional problems. This is because their method is computationally expensive and it is difficult to construct relationships between the numerical solution and the grid speed.

Saltzman and Brackbill [7-10] used variational principles to develop

a solution-adaptive, differential equation grid generation method. Their method is based on an idea proposed by Winslow [11]. Basically, a functional is constructed by summing three integrals representing stretching, smoothness and orthogonality of the boundary-fitted coordinate system in the physical domain. The minimization of that functional results in a set of elliptic partial differential equations whose solution gives the boundary-fitted coordinate system. The elliptic nature of the partial differential equations representing the coordinate transformation ensures a one-to-one mapping and generates a smooth and nearly orthogonal grid system. Saltzman and Brackbill applied their scheme to solve many different problems including an inviscid supersonic flow. In this problem, gradients were used to determine the stretching of the boundary-fitted coordinate system in order to concentrate grid points in regions with shock waves. The disadvantage of this method is that it is computationally expensive, since elliptic partial differential equations must be solved. This is especially true for unsteady, three-dimensional problems, in which a different grid system has to be generated at each time step.

Bell and Shubin [12] developed a solution-adaptive grid generation technique similar to that developed by Saltzman and Brackbill, but they only applied their method to analyze one-dimensional problems. Within the functional to be minimized by variational techniques, they introduced a spatial derivative term and a time derivative term to control the position of the grid points in the physical domain at each time step. Unfortunately, it is very difficult to construct the time derivative term in the functional to be minimized. Bell and Shubin

concluded after several applications of their method that, in comparison with non-adaptive grid generations techniques, their method expended a lot of work for only a modest increase in accuracy.

Rai and Anderson [13] developed a very interesting solution-adaptive grid generation method. In their method, each grid point has associated with it an attraction or repulsion depending upon the difference between the error of the numerical solution at each grid point and the average error over all grid points. The spacings between grid points are decreased or increased in proportion to those differences and the movements of the grid points are attenuated by an inverse power of the grid spacing in the physical domain. The magnitude of the first or second derivative of the solution was used as a measure of the error in the numerical solution. In this method, smoothness and orthogonality of the grid system were not controlled. Therefore, in order to prevent oscillations and distortions in the numerical solutions, this method imposes limitations on the movement of the grid points in the physical domain [14].

In many fluid flow problems, disparate length scales in the solution occur only in one coordinate direction. For those problems the grid spacing along each coordinate direction can be controlled independently. Dwyer et al. [15] have developed a method to control the distribution of grid points along one coordinate direction in the physical domain by using a positive weighting function. Such positive weighting function was a linear combination of first and second derivatives of the solution with respect to the chosen coordinate. The method has been applied successfully to solve many problems in heat

transfer and in two-dimensional flame propagation problems. The solution-adaptive, grid system generated by this method was found to improve the accuracy of numerical solutions considerably without the need to increase the number of grid points or the order of accuracy of the finite-differences equations. Dwyer et al. also noted that their solution-adaptive, grid generation method can remove oscillations often encountered in numerical solutions.

Gnoffo [16,17] developed a solution-adaptive, grid generation method based on tension spring analogy to redistribute grid points along one coordinate direction. In this method, the weighting function used was gradients of the numerical solution which are treated as spring constants in a system linking the grid points over the entire physical domain. To increase the number of grid points in certain regions, the spring constants were made larger in those regions, while the potential energy of the spring system was minimized. This method was applied with success to compute solutions to hypersonic viscous flow problems. These applications showed that adaptation along only one coordinate direction was sufficient to move the grid points toward regions where they are needed the most to resolve the physics of the problem being analyzed.

Nakahashi and Deiwert [18,19] used the approach of Gnoffo to develop a similar solution-adaptive, grid generation method. They extended the spring analogy to solve multidimensional fluid flow problems by applying successive adaptation in each coordinate direction. The smoothness was controlled by introducing constraints to ensure grid spacings do not change too fast at each time step and the orthogonality was controlled by introducing torsional springs between grid points in the physical

domain. They applied their method to generate a grid system in which grid points were adapted independently in each coordinate direction to study transonic flows past an airfoil. The density gradient was used to adapt the grid system in the streamwise direction for the purpose of resolving shock waves. In the direction normal to the airfoil surface, the component of momentum in that direction was used to adapt the grid system in order to resolve the physics in the boundary layer region. The advantage of this method is that the grid system is adapted independently in each coordinate direction and this permits the use of marching techniques. Thus, this method is much more efficient than methods requiring solution of elliptic partial differential equations for generating grid systems.

Smooke and Koszykowski [20] developed an adaptive grid generation method for solving one-dimensional initial-boundary value problems in which the number of grid points was adjusted to equidistribute a positive weighting function. The smoothness was achieved by imposing constraints such that the ratio between two adjacent grid spacings was less than a constant, given a priori, and greater than the inverse of that constant. The method required interpolation of the solution because the number of grid points are not the same at each time level and, sometimes, this was found to cause problems. The method was applied successfully to solve several combustion problems [21-22].

1.3 Objectives

Even though considerable progress has been made in grid generation, much work remains to be done in developing those techniques. This is

especially true for fluid flow problems with deforming spatial domain and problems with highly disparate length scales in different part of the spatial domain that can move about in a unpredictable manner. The major objective of this investigation was to develop a versatile and efficient solution-adaptive grid generation technique that can be used in the analysis of such fluid flow problems using finite-difference algorithms.

The grid generation method developed in this investigation differs from those methods developed by previous investigators in the following ways:

1. A mathematical formulation using variational principles was applied to develop a new technique for generating stretching functions. The stretching functions generated by using this technique couple the distribution of grid points in the physical domain to the numerical solution as it is being computed.
2. A new technique was developed based on the stretching function mentioned above and any algebraic, non-adaptive grid generation technique (e.g., the Two-Boundary Technique) to create a solution adaptive, algebraic grid generation method.
3. A new technique was developed to ensure that the grid system in the physical domain is smooth and that the grid lines do not overlap. Here, two new concepts were introduced: (1) the weighting function was controlled by a smoothness factor and (2) a minimum grid spacing specified a priori was used to control the nonoverlapping of the grid lines in those problems with length scales which continuously get shorter as time progresses.

4. A new technique was developed to ensure that the grid system in the physical domain is nearly orthogonal. When the method developed was applied in conjunction with the Two-Boundary Technique, an analytical approach was developed to determine the coefficients for controlling the orthogonality at the boundaries.

The solution-adaptive, algebraic grid generation method developed in this investigation is described in detail in Chapters II to V and is presented in the following order. First, the technique used to transform the governing equations from the coordinates of the physical domain to the boundary-fitted coordinates of the computational domain is described. Next, the procedure by which variational principles are used to develop stretching functions which couple the distribution of grid points to the numerical solution as it is being computed is presented. Then, a description is given on how stretching functions based on variational principles are used to construct a solution-adaptive grid generation method for one-dimensional problems. Afterwards, how the solution-adaptive grid generation technique for one-dimensional problems can be applied to multidimensional problems is described. Finally, the solution-adaptive grid generation method developed is used with the Two-Boundary Technique to generate grid points inside an arbitrarily shaped spatial domain.

Applications of the grid generation method developed in this investigation are described in Chapter VI to VIII and the conclusions as well as recommended future works are given in Chapter IX.

CHAPTER II

COORDINATE TRANSFORMATION AND METRIC COEFFICIENTS

In this chapter, the technique used to transform the governing equations from the coordinates of the physical domain to the boundary-fitted coordinates of the computational domain is described. Also, the technique for determining the metric coefficients for a two-dimensional coordinate transformation is presented.

Grid generation invariably involves the development of a boundary-fitted coordinate system which is said to be determined once the coordinate transformation between the coordinate system of the physical domain and the boundary-fitted coordinate system of the computational domain is known. If x-y-t is the coordinate system of the physical domain and ξ-η-τ is the boundary-fitted coordinate system of the computational domain, then the required coordinate transformation has the following form:

$$\begin{aligned} x &= x (\xi , \eta , \tau) \\ y &= y (\xi , \eta , \tau) \\ t &= t (\tau) \end{aligned} \tag{2.1}$$

Each grid point in the physical domain must correspond to only one grid point in the computational domain and vice versa. This means that

the coordinate transformation between the coordinate system of the physical domain and the coordinate system of computational domain must be one to one. The necessary and sufficient condition for a mapping to be one to one is that the Jacobian of the coordinate transformation be nonzero [23]. For the mapping between the two coordinate systems expressed by Eq. (2.1), the Jacobian is given by

$$J = \frac{\partial (x , y , t)}{\partial (\xi , \eta , \tau)} = \begin{vmatrix} x_\xi & x_\eta & x_\tau \\ y_\xi & y_\eta & y_\tau \\ 0 & 0 & t_\tau \end{vmatrix} =$$

$$= t_\tau (x_\xi y_\eta - x_\eta y_\xi) \qquad (2.2)$$

where the subscripts denote partial derivatives.

Since the Jacobian of the coordinate transformation cannot be zero, the inverse of the coordinate transformation exists and has the following form:

$$\begin{aligned} \xi &= \xi (x , y , t) \\ \eta &= \eta (x , y , t) \\ \tau &= \tau (t) \end{aligned} \qquad (2.3)$$

Figure 2.1 shows the transformation of the physical domain in a x-y-t coordinate system onto the computational domain in a ξ-η-τ boundary-fitted coordinate system. The grid points in the computational

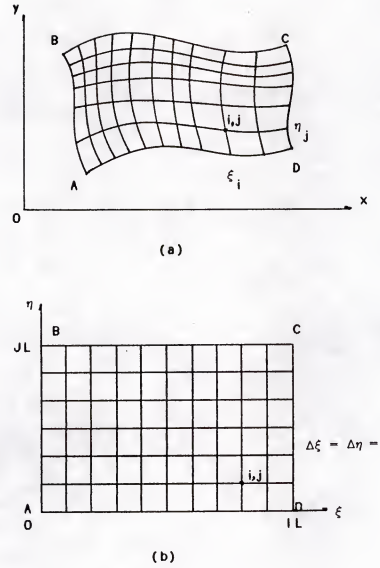


Figure 2.1 Transformation of (a) a physical domain to (b) a computational domain

domain are defined by the intersection of equally spaced coordinate lines. After the locations of grid points in the computational domain are known, the locations of grid points in the physical domain can be determined by using Eq. (2.1).

Once the coordinate transformation between the coordinate systems of the physical and the computational domains has been determined, the governing equations for the problem to be analyzed must be transformed so that ξ, η and τ become the independent variables instead of x, y and t. The transformation of the governing equations is accomplished in two steps. First, the relationship between partial derivatives with respect to x, y and t and partial derivatives with respect to ξ, η and τ must be established. Second, the partial derivatives with respect to x, y and t appearing in the governing equations must be replaced by the partial derivatives with respect to ξ, η and τ.

Any partial derivative of a dependent variable f with respect to x, y and t can be expressed in terms of ξ, η and τ by using the coordinate transformation given by Eq. (2.1) and the chain rule as follows:

$$\begin{Bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \\ \frac{\partial f}{\partial t} \end{Bmatrix} = \begin{bmatrix} \xi_x & \eta_x & 0 \\ \xi_y & \eta_y & 0 \\ \xi_t & \eta_y & \tau_t \end{bmatrix} \begin{Bmatrix} \frac{\partial f}{\partial \xi} \\ \frac{\partial f}{\partial \eta} \\ \frac{\partial f}{\partial \tau} \end{Bmatrix} \tag{2.4}$$

In the equation above, the elements of the matrix are known as

metric coefficients of the coordinate transformation. The metric coefficients depend only on the nature of the coordinate transformation given by Eq. (2.1) and do not depend on the physics of the problem being investigated. Equation (2.4) can be written in a vectorial form as $\vec{f} = [A]\,\vec{g}$ where [A] represent the matrix of the metric coefficients of the coordinate transformation.

Similarly, by using the inverse of the coordinate transformation given by Eq. (2.3), the following relation can be obtained:

$$\begin{Bmatrix} \frac{\partial f}{\partial \xi} \\ \frac{\partial f}{\partial \eta} \\ \frac{\partial f}{\partial \tau} \end{Bmatrix} = \begin{bmatrix} x_\xi & y_\xi & 0 \\ x_\eta & y_\eta & 0 \\ x_\tau & y_\tau & t_\tau \end{bmatrix} \begin{Bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \\ \frac{\partial f}{\partial t} \end{Bmatrix} \tag{2.5}$$

which in a vectorial form is expressed as $\vec{g} = [B]\,\vec{f}$.

Using Eqs. (2.4) and (2.5), namely, $\vec{f} = [A]\,\vec{g}$ and $\vec{g} = [B]\,\vec{f}$, gives $[A] = [B]^{-1}$. Since the elements of matrix [A] should be identical to the elements of the inverse of the matrix [B], the metric coefficients can be written as follows:

$$\xi_x = \frac{1}{J}\begin{vmatrix} y_\eta & y_\tau \\ 0 & t_\tau \end{vmatrix} \qquad \xi_y = \frac{-1}{J}\begin{vmatrix} x_\eta & x_\tau \\ 0 & t_\tau \end{vmatrix} \tag{2.6,7}$$

$$\eta_x = \frac{-1}{J}\begin{vmatrix} y_\xi & y_\tau \\ 0 & t_\tau \end{vmatrix} \qquad \eta_y = \frac{1}{J}\begin{vmatrix} x_\xi & x_\tau \\ 0 & t_\tau \end{vmatrix} \tag{2.8,9}$$

$$\xi_t = \frac{1}{J}\begin{vmatrix} x_\eta & x_\tau \\ y_\eta & y_\tau \end{vmatrix} \qquad \eta_t = \frac{-1}{J}\begin{vmatrix} x_\xi & x_\tau \\ y_\xi & y_\tau \end{vmatrix} \tag{2.10,11}$$

$$\tau_t = \frac{1}{J}\begin{vmatrix} x_\xi & x_\eta \\ y_\xi & y_\eta \end{vmatrix} = \frac{1}{t_\tau} \tag{2.12}$$

Substituting Eqs. (2.6) through (2.12) into Eq. (2.4) gives the desired relationships between the partial derivatives with respect to x, y and t and the partial derivatives with respect to ξ, η and τ. These relationships are used to replace the partial derivatives with respect to x, y, and t appearing in the governing equations by partial derivatives with respect to ξ, η and τ. After the governing equations have been expressed in terms of the boundary-fitted coordinates, the governing equations are transformed to finite-difference equations which are used to analyze the problem in the computational domain.

Here, it is noted that the metric coefficients given by Eqs. (2.6) through (2.12) are only valid for unsteady, two-dimensional problems. However, the procedure used to derive those equations can be applied in a straightforward manner to derive the metric coefficients for unsteady, three-dimensional problems [3].

CHAPTER III

THE SAAGG METHOD FOR ONE-DIMENSIONAL PROBLEMS

In this chapter, a solution-adaptive, algebraic grid generation (SAAGG) method for one-dimensional problems is developed. The method is presented in the following sequence. First, a new class of stretching function based on variational principle is presented. Second, how the stretching function, thus developed, is used to create a solution-adaptive grid generation method is described. Third, a smoothness factor is defined to control the smoothness of the grid system being generated. Finally, the algorithm for this grid generation method is summarized.

3.1 Variational Principles and Stretching Functions

Stretching functions have been widely used in grid generation techniques to redistribute grid points in the physical domain to improve accuracy of numerical solution by concentrating grid points where most needed [24]. In order to construct appropriate stretching functions, it is necessary to know qualitatively the physics of the problem being analyzed. For unsteady fluid flow problems in which disparate length scales in different parts of the physical domain can move about in an unpredictable manner, the required qualitative information is unknown and it is difficult to construct appropriate stretching functions. For such problems, the stretching function should be coupled to the

numerical solution being computed and it also should be able to redistribute any existing grid system to minimize numerical errors. A stretching function with such features, derived from a variational technique, is developed in this chapter for one-dimensional problems.

In order to generate a stretching function based on a variational technique, some functional to be minimized must be formed first. In this investigation, that functional was taken to be the integral of the product of some property of the grid (G), related to the grid spacing in the physical domain, and a positive weighting function (W), related to the error in the numerical solution. The minimization of this functional guarantees that the grid points in the physical domain will be distributed so that the grid spacing will be small where the weighting function is large, and vice versa. The reason W was squared in the functional below is to ensure that the weighting function is always positive. Thus, the functional to be minimized is expressed by

$$I = \int W^2(x)\, G(\bar{x})\, d\xi \qquad (3.1.1)$$

In the equation above, ξ is the coordinate of computational domain, x is the coordinate of the physical domain before stretching, and $\bar{x}$ is the coordinate of the physical domain after stretching. Both x and $\bar{x}$ are functions of ξ. The property of grid (G) is defined below and the weighting function (W) is defined in Section 3.2.1.

The property of the grid (G) in the functional to be minimized was chosen to be the square of the grid spacing in the physical domain after

stretching. If the grid spacing in the computational domain is taken to be unity (i.e. $\Delta\xi=1$), then the grid spacing in the physical domain ($\Delta\bar{x}$) can be expressed as follows:

$$\bar{x}_\xi \approx \frac{\Delta\bar{x}}{\Delta\xi} \qquad \text{or} \qquad \Delta\bar{x} \approx \bar{x}_\xi \tag{3.1.2}$$

Thus, the property of the grid (G) can be written as follows:

$$G = \Delta\bar{x}^2 = \bar{x}_\xi^2 \tag{3.1.3}$$

The stretching function (K) to be developed in this investigation relates the coordinates of the physical domain before and after stretching as follows:

$$\bar{x} = K(x)\, x \tag{3.1.4}$$

By using Eqs. (3.1.3) and (3.1.4), the functional to be minimized by the variational technique expressed by Eq. (3.1.1) can now be written as shown below:

$$I = \int \left(W(x)\, [K(x)x]_\xi\right)^2 d\xi \tag{3.1.5}$$

The partial derivative term appearing in Eq. (3.1.5) is obtained by differentiation as follows:

$$[K(x)x]_\xi = K(x) \frac{\partial x}{\partial \xi} + x \frac{\partial K}{\partial \xi}$$

$$= K(x) \frac{\partial x}{\partial \xi} + x \frac{dK}{dx} \frac{\partial x}{\partial \xi}$$

$$= \left[K(x) + x \frac{dK}{dx} \right] x_\xi \qquad (3.1.6)$$

Substitution of Eq. (3.1.6) into Eq. (3.1.5) gives

$$I = \int [W(x)\, M(x)\, x_\xi]^2 d\xi \qquad (3.1.7)$$

where

$$M(x) = K(x) + x \frac{dK}{dx} \qquad (3.1.8)$$

Equation (3.1.7) can be written more compactly by letting

$$N(x) = W(x)\, M(x) \qquad (3.1.9)$$

so that the functional to be minimized becomes

$$I = \int [\, N(x)\, x_\xi \,]^2 \, d\xi \tag{3.1.10}$$

The problem of minimizing the integral above can be solved by a variational technique and the solution is the Euler-Lagrange equation associated with that integral. The Euler-Lagrange equation, derived in Appendix A, is as follows:

$$\frac{\partial F}{\partial x} - \frac{d}{d\xi}\left(\frac{\partial F}{\partial x_\xi}\right) = 0 \tag{3.1.11}$$

where

$$F = [\, N(x)\, x_\xi \,]^2 \tag{3.1.12}$$

$$\frac{\partial F}{\partial x} = 2\, N(x)\, x_\xi^2 \frac{dN}{dx} \tag{3.1.13}$$

and

$$\frac{\partial F}{\partial x_\xi} = 2\, N(x)^2 x_\xi \tag{3.1.14}$$

Substituting Eqs. (3.1.13) and (3.1.14) into Eq. (3.1.11) gives

$$N(x)\, x_\xi^2 \frac{dN}{dx} - \frac{d}{d\xi}[\, N(x)^2 x_\xi \,] = 0 \tag{3.1.15}$$

which after simplification yields the following ordinary differential equation:

$$\frac{d}{dx} [\, N(x)\, x_\xi \,] = 0 \tag{3.1.16}$$

By integrating the above equation and using Eqs. (3.1.8) and (3.1.9), Eq. (3.1.16) becomes

$$\frac{d}{dx} [\, K\, x \,] = C_2 \frac{\xi_x}{W(x)} \tag{3.1.17}$$

where C_2 is a constant of integration. Equation (3.1.17) can be integrated again to yield

$$K(x) = \frac{1}{x} [\, C_1 + C_2 \int \frac{\xi_x}{W(x)}\, dx \,] \tag{3.1.18}$$

producing another constant of integration, C_1.

The constants of integration, C_1 and C_2, can be evaluated by requiring that the coordinates before and after stretching have the same value at each boundary of the physical domain. By assuming the boundaries of the physical domain to be at L_1 and L_2, Eq. (3.1.4) implies that

$$K(x=L_1) = 1 \tag{3.1.19}$$

and

$$K(x=L_2) = 1 \tag{3.1.20}$$

By using Eqs. (3.1.19) and (3.1.20), Eq. (3.1.18) yields the following system of equations:

$$\begin{cases} L_1 = C_1 + C_2 \left[\int \frac{\xi_x}{W(x)} dx \right]_{x=L_1} & (3.1.21) \\ L_2 = C_1 + C_2 \left[\int \frac{\xi_x}{W(x)} dx \right]_{x=L_2} & (3.1.22) \end{cases}$$

Solving the above system of equations for the constants, C_1 and C_2, yields

$$C_1 = L_1 - (L_2-L_1) \frac{\left[\int \frac{\xi_x}{W(x)} dx \right]_{x=L_1}}{\int_{L_1}^{L_2} \frac{\xi_x}{W(x)} dx} \tag{3.1.23}$$

and

$$C_2 = \frac{L_2 - L_1}{\int_{L_1}^{L_2} \frac{\xi_x}{W(x)} dx} \tag{3.1.24}$$

Finally, substituting Eqs. (3.1.23) and (3.1.24) into Eq. (3.1.18) gives the desired expression for the stretching function which is

$$K(x) = \frac{L_1 + (L_2 - L_1)\dfrac{\displaystyle\int_{L_1}^{x} \frac{\xi_x}{W(x)}\,dx}{\displaystyle\int_{L_1}^{L_2} \frac{\xi_x}{W(x)}\,dx}}{x} \qquad (3.1.25)$$

The integrals in Eq. (3.1.25) can be evaluated numerically or analytically to determine the value of K(x) at each grid point. Once K(x) has been obtained, it is substituted into Eq. (3.1.4) to give the location of grid points in the physical domain after stretching.

Application of this technique for generating stretching functions is presented in Chapter VI where several examples are given.

3.2 A Solution-Adaptive Grid System

The stretching function presented in the previous section can couple the distribution of grid points to the numerical solution of the problem that is being analyzed through the weighting function. Thus, these stretching functions can be employed to develop a solution-adaptive, grid generation method to be used for those problems with disparate length scales in different parts of the spatial domain which can move about in an unpredictable manner. This solution-adaptive, grid generation method has the advantage of being an algebraic method because

partial differential equations do not need to be solved in order to determine the locations of the grid points.

As noted earlier, in order for a grid generation method to be useful, it should satisfy constraints on smoothness and orthogonality in addition to stretching. For one-dimensional problems, only stretching and smoothness need to be considered. In this section, stretching and smoothness of a one-dimensional grid system are discussed. Stretching, smoothness and orthogonality for multidimensional grid systems are discussed in the next chapter.

The stretching function represented by Eq. (3.1.25) can be substituted into Eq. (2.1) to give the coordinate transformation necessary to generate grid points in the physical domain. By noting that

$$dx = x_\xi \, d\xi = \frac{1}{\xi_x} \, d\xi \tag{3.2.1}$$

the relationship for the coordinate transformation can be represented by the following equation:

$$\bar{x} = L_1 + (L_2 - L_1) \frac{\int_1^\xi \frac{1}{W(x)} \, d\xi}{\int_1^{IL} \frac{1}{W(x)} \, d\xi} \tag{3.2.2}$$

where

$$\bar{x} = x(\xi, \bar{\tau}^{n+1}) \tag{3.2.3}$$

and

$$x = x(\xi, \tau^{n+1}) \tag{3.2.4}$$

In the above equations, x is the coordinate of the grid points in the physical domain at time level n+1 before stretching; $\bar{x}$ is the coordinate of grid points at time level n+1 after stretching; L_1 and L_2 are the boundaries of the spatial domain; and 1 and IL are the boundaries of the computational domain.

The locations of the grid points in the physical domain, determined by Eq. (3.2.2), can be coupled to the numerical solution as it is being computed through the weighting function (W). Such weighting function which makes the method to be a solution-adaptive grid generation method is presented in the next section.

3.2.1 Weighting Function

The weighting function appearing in Eq. (3.2.2) should be related to the numerical solution to make the method solution-adaptive. In this investigation, the weighting function was chosen as a function of the gradient of some dependent variable of the problem being solved. This is because, in general, numerical errors are larger in those regions of the physical domain where the gradients are larger [25]. However, letting the weighting function equal to that gradient causes two problems. First, when the gradient is zero, Eq. (3.2.2) is singular. Second, when the gradient is very large, almost all of the grid points will be

clustered in the vicinity of the large gradient so that the grid system will not be smooth. To overcome the first problem, a positive constant, in this investigation chosen to be unity, was added to the absolute value of the gradient of the solution. To overcome the second problem, the constant and the absolute value of the gradient were raised to some power SF. Thus, the weighting function used in this investigation has the following form:

$$W = \omega^{SF} = (1 + | U_x |)^{SF} \tag{3.2.5}$$

The power SF is referred to as the smoothness factor and controls the value of the weighting function in order to ensure that the grid system generated is smooth. The procedure for determining the smoothness factor is described in the next section.

3.2.2 Smoothness Factor

The spacing between two adjacent grid points in the physical domain can be obtained from Eq. (3.2.2) as follows:

$$\Delta x_i = x_{i+1} - x_i = (L_2 - L_1) \frac{\displaystyle\int_{\xi_1}^{\xi_{i+1}} \frac{1}{[\omega]^{SF}} d\xi}{\displaystyle\int_{1}^{IL} \frac{1}{[\omega]^{SF}} d\xi} \tag{3.2.6}$$

The integrals in Eq. (3.2.6) can be evaluated numerically by using the backward-step method as shown below:

$$\Delta x_i = (L_2 - L_1) \frac{\left| \frac{1}{\omega_i} \right|^{SF}}{\sum_{j=2}^{IL} \left| \frac{1}{\omega_j} \right|^{SF}} \tag{3.2.7}$$

Equation (3.2.7) can be written more compactly as follows:

$$\Delta x_i = \frac{L_2 - L_1}{\sum_{j=2}^{IL} \left| \frac{\omega_i}{\omega_j} \right|^{SF}} \tag{3.2.8}$$

Here, it is noted that higher-order accurate integration methods could have been used instead of the backward-step method. However, for the purpose of grid generation, the backward-step method was found to give results as good as those obtained by using Simpson's rule.

The grid spacings in the physical domain can be no greater than Δx_{max}, and no smaller than Δx_{min}. Both Δx_{max} and Δx_{min} are specified a priori. Hence, since ω_i is a minimum when Δx_i is a maximum and vice versa, Eq. (3.2.8) gives the following relations:

$$\sum_{j=2}^{IL} \left| \frac{\omega_{max}}{\omega_j} \right|^{SF} = \frac{L_2 - L_1}{\Delta x_{min}} \tag{3.2.9}$$

and

$$\sum_{j=2}^{IL} \left| \frac{\omega_{min}}{\omega_j} \right|^{SF} = \frac{L_2 - L_1}{\Delta x_{max}} \tag{3.2.10}$$

Equations (3.2.9) and (3.2.10) can be rewritten as shown below:

$$(\omega_{max})^{SF} \sum_{j=2}^{IL} \left| \frac{1}{\omega_j} \right|^{SF} = \frac{L_2 - L_1}{\Delta x_{min}} \tag{3.2.11}$$

$$(\omega_{min})^{SF} \sum_{j=2}^{IL} \left| \frac{1}{\omega_j} \right|^{SF} = \frac{L_2 - L_1}{\Delta x_{max}} \tag{3.2.12}$$

The solution of the above system of equations gives the following expression for the smoothness factor (SF):

$$SF = \frac{\ln \dfrac{\Delta x_{max}}{\Delta x_{min}}}{\ln \dfrac{\omega_{max}}{\omega_{min}}} \tag{3.2.13}$$

The above expression relates SF to two ratios, namely $\omega_{max}/\omega_{min}$ and $\Delta x_{max}/\Delta x_{min}$. A numerical value for SF can be obtained once these two ratios are specified.

The ratio $\omega_{max}/\omega_{min}$ is determined by using Eq. (3.2.5) and by examining the numerical solution (U) being computed at time level n, as shown below.

$$\frac{\omega_{max}}{\omega_{min}} = \frac{(1 + \max|U_x|)^n}{(1 + \min|U_x|)^n} \tag{3.2.14}$$

In the above equation, $\max|U_x|^n$ and $\min|U_x|^n$ denote the maximum and the minimum absolute values of U_x in the physical domain at time level n, respectively.

Specification of the ratio $\Delta x_{max}/\Delta x_{min}$ depends on the problem being analyzed. If the maximum gradient of the problem being solved does not change in magnitude as time progresses, then the ratio $\Delta x_{max}/\Delta x_{min}$ can be set equal to a constant which is the same for every time step; i.e.,

$$\frac{\Delta x_{max}}{\Delta x_{min}} = \text{constant} \tag{3.2.15}$$

For problems in which the maximum gradient of the problem being solved increases as time progresses, the ratio $\Delta x_{max}/\Delta x_{min}$ should be changed from time step to time step. Here, that ratio is specified as follows:

$$\left|\frac{\Delta x_{max}}{\Delta x_{min}}\right|^{n+1} = \left|\frac{\Delta x_{max}}{\Delta x_{min}}\right|^{n} \frac{\left|\frac{\omega_{max}}{\omega_{min}}\right|^{n+1}}{\left|\frac{\omega_{max}}{\omega_{min}}\right|^{n}} \tag{3.2.16}$$

where $(\omega_{max})^{n+1}$ and $(\omega_{min})^{n+1}$ are obtained from the numerical solution U^{n+1} by using a grid system obtained at time level n+1 before stretching.

With the ratios $\omega_{max}/\omega_{min}$ and $\Delta_{max}/\Delta_{min}$ specified in the manner described above, Eq. (3.2.13) readily gives a numerical value for SF. Once SF has been thus determined, the location of each grid point can be determined by using Eq. (3.2.2) as follows:

$$\bar{x}_i^{n+1} = L_1 + (L_2 - L_1) \frac{\int_1^{\xi_i} \frac{1}{[\omega^{SF}]^n} d\xi}{\int_1^{IL} \frac{1}{[\omega^{SF}]^n} d\xi} \tag{3.2.17}$$

where ω^{SF} is obtained from Eq. (3.2.5).

The grid generation method given by Eq. (3.2.17) may produce grid spacings that are smaller than that needed to obtain solutions of the desired accuracy. In order to prevent this situation, the smallest grid spacing allowed (Δx_{crt}) must be specified a priori. The precise value of Δx_{crt} depends on the physics of the problem being analyzed and the

accuracy desired in the computed numerical solution. If Δx_{min} computed by using Eq. (3.2.7) is less than Δx_{crt}, then the smoothness factor (SF) is obtained iteratively from the following nonlinear equation:

$$\Delta x_{crt} \sum_{j=2}^{IL} \left| \frac{1}{[\omega_j]^k} \right|^{SF} = \frac{L_2 - L_1}{|[\omega_{max}]^{k+1}|^{SF}} \tag{3.2.18}$$

where k denotes iteration. The above equation was obtained from Eq. (3.2.9) by setting Δx_{min} to Δx_{crt}.

This concludes the description of the solution-adaptive grid generation method for one-dimensional problems. The details of the implementation of this method are described in the next section.

3.3 Algorithm for One-Dimensional Problems

This section presents the algorithm for generating solution-adaptive grid systems for one-dimensional problems in which the two boundaries of the physical domain can move or be stationary. The algorithm which will map the grid points between the physical domain in x-t coordinate system and the computational domain in the ξ-τ coordinate system is as follows:

(1) Transform the governing equations from the coordinate system of the physical domain to the coordinate system of the computational domain.

(2) Apply any suitable finite-difference method to express the governing equations as a set of finite difference equations (FDEs).

(3) Specify the duration of interest (i.e. $0 \le t \le T$) and set the time level n to zero which corresponds to time t=0.

(4) Specify the number of grid points (IL) to be employed. IL depends on the accuracy desired in the numerical solutions and on the computer resources available. With IL specified, Eq. (3.2.2) implies that $x=L_1$ corresponds to $\xi=1$ and $x=L_2$ corresponds to $\xi=IL$ as expressed by the following equations:

$$x\ (\xi=1) = L_1 \tag{3.3.1}$$

$$x\ (\xi=IL) = L_2 \tag{3.3.2}$$

(5) Specify the locations of the grid points in the computational domain. As noted before, the grid points in the computational domain are always equally spaced and the grid spacing is unity ($\Delta\xi=1$) so that the locations of the grid point are given by

$$\xi_i = i\ , \qquad i = 1,2,\ldots,IL \tag{3.3.3}$$

(6) Specify the locations of the grid points in the physical domain at the initial time level (n=0). The initial grid system specified may consist of equally spaced or nonequally spaced grid points. If the grid points are equally spaced, then the locations of the grid point are given by

$$x_i = L_1 + \frac{L_2 - L_1}{IL - 1}(\xi_i - 1) \qquad (3.3.4)$$

$$i = 1,2,\ldots,IL$$

where ξ_i is given by Eq. (3.3.3). If it is desired to use nonequally spaced grid points for the initial grid system because of the nature of the initial conditions, then Eq. (3.2.17) can be used to generate such a grid system as shown below:

$$x_1 = L_1 \qquad (3.3.5)$$

$$\overset{n}{x}_i = \overset{n}{L}_1 + (\overset{n}{L}_2 - \overset{n}{L}_1)\frac{\int_1^{\xi_i} \frac{1}{\omega^{SF}}\,d\xi}{\int_1^{IL} \frac{1}{\omega^{SF}}\,d\xi}$$

$$\approx \overset{n}{L}_1 + (\overset{n}{L}_2 - \overset{n}{L}_1)\frac{\sum_{m=2}^{i} \frac{1}{\omega(x_m)^{SF}}}{\sum_{m=2}^{IL} \frac{1}{\omega(x_m)^{SF}}} \qquad i=2,4,\ldots,IL \qquad (3.3.6)$$

where ξ_i is given by Eq. (3.3.3) and

$$\omega(x_m) = 1 + \left| \frac{\partial U(x_m)}{\partial x} \right|$$

$$\approx 1 + \left| \frac{U(x_{m+1}) - U(x_{m-1})}{x_{m+1} - x_{m-1}} \right| \qquad (3.3.7)$$

In the above equations, U is the initial condition; x_m is given by Eq. (3.3.4); and SF is determined by using either Eq.(3.2.13) or Eq. (3.2.18).

(7) Evaluate the metric coefficients numerically from the following equations:

$$\xi^n_{x_i} = \frac{1}{x^n_{\xi_i}} = \frac{2\Delta\xi_i}{x^n_{i+1} - x^n_{i-1}} \qquad (3.3.8)$$

and

$$\xi^n_{t_i} = \frac{2\Delta\xi_i}{x^n_{i+1} - x^n_{i-1}} \frac{x^{n+1}_i - x^n_i}{\Delta t_i} \qquad (3.3.9)$$

where

$$x^{n+1}_i = \frac{L^{n+1}_2 - L^{n+1}_1}{L^n_2 - L^n_1} (x^n - L^n_1) + L^{n+1}_1 \qquad (3.3.10)$$

In the above equations x^n_i is obtained in step 6.

(8) Use the FDEs derived in step 2 to obtain a "tentative" solution of the problem at the next level by using the grid system generated in step 6.

(9) Determine the locations of the grid points at next time level by using Eqs. (3.3.6) and (3.3.7) in which U(x) is the solution obtained in step 8.

(10) Recalculate the metric ξ_t^n by using Eq. (3.3.9) with x_i^{n+1} obtained in step 9.

(11) Use the FDEs with the metric coefficients given by Eqs. (3.3.8) and (3.3.9) to obtain the "corrected" solution at the next time level.

(12) Replace n by n+1 and calculate the time at that time level by using the expression

$$t^{n+1} = \sum_{m=0}^{n+1} \Delta t_m \tag{3.3.11}$$

(13) Repeat steps 7 to 11 if $t^{n+1} < T$. If $t^{n+1} \geq T$, then stop the algorithm.

Several examples illustrating how the solution-adaptive grid generation method developed in this chapter can be applied to analyze fluid flow problems are presented in Chapter VII.

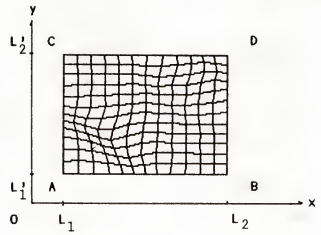
CHAPTER IV

THE SAAGG METHOD FOR 2-D, RECTANGULARLY SHAPED DOMAINS

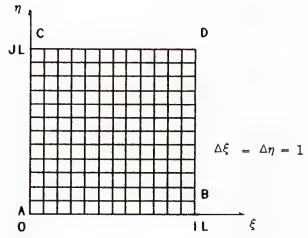
In this chapter, a solution-adaptive, algebraic grid generation (SAAGG) method for two-dimensional, rectangularly shaped spatial domains is presented. This method is based on the one-dimensional, solution-adaptive grid generation method described in Chapter III. The two-dimensional, solution-adaptive grid generation method is presented in the following sequence. First, an overview of the method is described. Second, the techniques used to control the smoothness of the grid system and the nonoverlapping of the grid lines are presented. Finally, the automatic adjustments for controlling the orthogonality of the grid system are defined.

4.1 A Two-Dimensional, Solution-Adaptive Grid System

In order to explain the solution-adaptive, grid generation method for two-dimensional, rectangularly shaped spatial domains, consider mapping the physical domain shown in Fig. 4.1a in the x-y-t coordinate system to the computational domain in the ξ-η-τ coordinate system shown in Fig. 4.1b. The spatial domain shown in Fig 4.1a can undergo rectilinear deformations. Here, the coordinate transformation sought between the x-y-t and the ξ-η-τ coordinate systems is given by Eq. (2.1) and is repeated below for convenience.



(a)



(b)

Figure 4.1 Correspondence between the boundaries of the physical domain and the computational domain. (a) Rectangularly shaped physical domain (x-y-t). (b) Computational domain (ξ-η-τ).

$$x = x(\xi,\eta,\tau)$$
$$y = y(\xi,\eta,\tau) \qquad (2.1)$$
$$t = \tau$$

Implementation of the solution-adaptive, grid generation method for two-dimensional, rectangularly shaped spatial domains involves the following steps:

(1) Transform the governing equations from the coordinate system of the physical domain to the coordinate system of the computational domain.

(2) Apply any suitable finite-difference method to express the transformed governing equations as a set of finite-difference equations (FDEs).

(3) Decide the correspondence between the boundaries of the physical domain and the boundaries of the computational domain. In this investigation, this correspondence is shown in Fig. 4.1.

(4) Specify the number of grid lines IL in the ξ-direction and the number of grid line JL in the η-direction.

(5) Set the initial time level n equal to zero which corresponds to time t=0 and specify the duration of interest T (i.e. $0 \le t \le T$).

(6) Specify the locations of the grid points in the computational domain. Here, $\Delta\xi=1$ and $\Delta\eta=1$ so that

$$\xi_i = i \qquad i=1,2,\ldots,IL \qquad (4.1.1)$$

and

$$\eta_j = j \qquad j=1,2,\ldots,JL \qquad (4.1.2)$$

(7) Specify the locations of the grid points in the physical domain at the initial time level (n=0). The initial grid system may have grid lines that are equally spaced or nonequally spaced. If it is desired to have equally spaced grid lines, then the x- and y-coordinates of the grid points in the physical domain are given by the following expressions:

$$x_i^n = L_1^n + \frac{L_2^n - L_1^n}{IL - 1}(\xi_i - 1) \qquad (4.1.3)$$

$$i = 1,2,\ldots,IL$$

and

$$y_j^n = L_1'^n + \frac{L_2'^n - L_1'^n}{JL - 1}(\eta_j - 1) \qquad (4.1.4)$$

$$j = 1,2,\ldots,JL$$

If it is desired to have nonequally spaced grid lines, because of steep gradients appearing in the initial condition, then the x-coordinates along the j^{th} grid line are given by Eqs. (3.3.5) to (3.3.7). In those equations x_m is given by Eq. (4.1.3); the smoothness factor is taken to be unity (i.e. SF=1); and $U(x_m,x_i)$ is the initial condition of the problem. The y-coordinates of the grid points are obtained by using the procedure just described except replace x by y, i by j, L_1 by L_1', L_2 by L_2', and $U(x_m,y_j)$ by $U(x_i,y_m)$ with y_m given by Eq. (4.1.4).

(8) Evaluate the metric coefficients of the coordinate transforma-

tion by using FDEs obtained from Eqs. (2.6) to (2.12). These FDEs are derived by using second-order-accurate, central-difference formulas, similar to that shown for Eqs. (3.3.8) and (3.3.9). The coordinates x_{ij}^{n} and y_{ij}^{n} appearing in those FDEs describe the locations of the grid points at time level n. The coordinate x_{ij}^{n+1} appearing in those FDEs are given by the following equations:

$$x_{ij}^{n+1} = L_{ij}^{n+1} \qquad i=1 \tag{4.1.5}$$

$$x_{ij}^{n+1} = \frac{\Delta L_j^{n+1}}{\Delta L_j^{n}} \Delta S_{ij}^{n} + L_{ij}^{n+1} \tag{4.1.6}$$

$$i=2,3\ldots,IL$$

where

$$\Delta L_j^{n} = \sum_{m=2}^{IL} \left[\left(x_{m,j}^{n} - x_{m-1,j}^{n}\right)^2 + \left(y_{m,j}^{n} - y_{m-1,j}^{n}\right)^2 \right]^{1/2} \tag{4.1.7}$$

$$\Delta S_{ij}^{n} = \sum_{m=2}^{i} \left[\left(x_{m,j}^{n} - x_{m-1,j}^{n}\right)^2 + \left(y_{m,j}^{n} - y_{m-1,j}^{n}\right)^2 \right]^{1/2} \tag{4.1.8}$$

and

$$\Delta L_j^{n+1} \approx \frac{L_2^{n+1} - L_1^{n+1}}{L_2^{n} - L_1^{n}} \Delta L_j^{n} \tag{4.1.9}$$

The coordinates y_{ij}^{n+1} appearing in those FDEs are given by the following equations:

$$y_{ij}^{n+1} = L_{ij}^{,n+1} \qquad j=1 \tag{4.1.10}$$

$$y_{ij}^{n+1} = \frac{\Delta L_j^{,n+1}}{\Delta L_j^{,n}} \Delta S_{ij}^{n} + L_{ij}^{n+1} \tag{4.1.11}$$

$$j=2,3\ldots,JL$$

where

$$\Delta L_i^{,n} = \sum_{m=2}^{JL} \left[(x_{i,m}^{n} - x_{i,m-1}^{n})^2 + (y_{i,m}^{n} - y_{i,m-1}^{n})^2 \right]^{1/2} \tag{4.1.12}$$

$$\Delta S_{ij}^{,n} = \sum_{m=2}^{i} \left[(x_{i,m}^{n} - x_{i,m-1}^{n})^2 + (y_{i,m}^{n} - y_{i,m-1}^{n})^2 \right]^{1/2} \tag{4.1.13}$$

and

$$\Delta L_j^{,n+1} \approx \frac{L_2^{,n+1} - L_1^{,n+1}}{L_2^{,n} - L_1^{,n}} \Delta L_j^{,n} \tag{4.1.14}$$

In the above equations, ΔL_j^n is the length of the j^{th} grid line in the physical domain at the n^{th} time level; ΔS_{ij}^n is the length of the j^{th} grid line from i=1 to i; ΔL_j^{n+1} is an estimate of the length of the j^{th} grid line at the $(n+1)^{th}$ time level. The physical significance of $\Delta L_i^{,n}$, $\Delta S_{ij}^{,n}$ and $\Delta L_i^{,n+1}$ is similar to that explained for ΔL_j^n, ΔS_{ij}^n and ΔL_j^{n+1}.

(9) Apply the FDEs derived in step 2 to obtain a "tentative" solution of the problem being solved at the $(n+1)^{th}$ level by using the grid systems given by (x_{ij}^n, y_{ij}^n) and $(x_{ij}^{n+1}, y_{ij}^{n+1})$.

(10) Evaluate the smoothness factor by using the grid system given by $(x_{ij}^{n+1}, y_{ij}^{n+1})$ and the "tentative" solution obtained in step 9. This is very involved and will be explained in Section 4.2.

(11) Recalculate the locations of the grid points at $(n+1)^{th}$ time level by using the smoothness factor obtained in the previous step. The x-coordinates of the grid points along the j^{th} grid line at the $(n+1)^{th}$ time level are computed by using Eqs.(3.3.5) to (3.3.7). In these equations, $x_m=x_{mj}^{th}$; the smoothness factor is that evaluated at step 10; and $U(x_m)= U(x_{mj},y_{mj})$ where U is given by the "tentative" solution obtained in step 9. The y-coordinates of the grid points along the i^{th} grid line are obtained by using the procedure just described except replace x by y, i by j, L_1 by L_1', L_2 by L_2', and $U(x_{mj},y_{mj})$ by $U(x_{im},y_{im})$.

(12) Control the orthogonality of the grid system obtained in step 11. This is very involved and will be explained in Section 4.3.

(13) Recalculate the metric coefficients by using the equations described in step 8 with x_{ij}^{n+1} and y_{ij}^{n+1} obtained in step 12.

(14) Use the FDEs with the metric coefficients calculated in step 13 to obtain the "corrected" solution at the $(n+1)^{th}$ time level.

(15) Replace the time level n by n+1 and calculate the total time elapsed by using Eq. (3.3.11). Repeat steps 8 to 14 until the total time elapsed equals to the duration of interest specified in step 5.

This completes the description of the solution-adaptive, grid generation method for two-dimensional, rectangularly shaped spatial domains that can undergo rectilinear deformations. In the next two sections, the techniques used to control nonoverlapping of grid lines and the smoothness of the grid system as well as the procedure used to control the orthogonality of grid system are explained.

4.2 Nonoverlapping of Grid Lines and Smoothness

When using the solution-adaptive, grid generation method presented in the last section to generate grid points inside two-dimensional spatial domains, it is important to control not only the clustering of grid points in regions where most needed, but also the nonoverlapping of the grid lines and the smoothness of the grid system being generated. In this investigation, nonoverlapping of grid lines and smoothness were controlled by the smoothness factor described in Section 3.2.2. and by monitoring a cell area associated with each grid point.

The cell area, Ag, associated with each grid point can be defined by the magnitude of the cross product of the vectors tangent to the curves ξ and η at grid point P shown in Fig 4.2 and represented by the following equation:

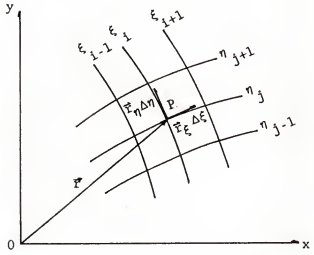


Figure 4.2 Definition of the vectors tangent to the curves ξ and η at grid point P.

$$Ag \approx \left| \frac{\partial \vec{r}}{\partial \eta} \Delta\eta \times \frac{\partial \vec{r}}{\partial \xi} \Delta\xi \right|$$

$$Ag \approx \left| \left(\frac{\partial x}{\partial \eta} \Delta\eta \vec{i} + \frac{\partial y}{\partial \eta} \Delta\eta \vec{j} \right) \times \left(\frac{\partial x}{\partial \xi} \Delta\xi \vec{i} + \frac{\partial y}{\partial \xi} \Delta\xi \vec{j} \right) \right| \qquad (4.2.1)$$

which after efetuating the vectorial operations reduces to

$$Ag \approx \frac{\partial x}{\partial \xi} \frac{\partial y}{\partial \eta} - \frac{\partial x}{\partial \eta} \frac{\partial y}{\partial \xi} \qquad (4.2.2)$$

when $\Delta\eta = \Delta\xi = 1$. In Eq. (4.2.1) $\vec{i}$ and $\vec{j}$ are unit vectors point in the x- and y-direction, repectively.

The equation above is the Jacobian of the coordinate transformation given by Eq. (2.2) when $t=\tau$. This fact will be useful because the Jacobian has already been calculated while computing the "tentative" solution. The Jacobian can be evaluated numerically by either one of the following two ways:

$$Ag_{ij} = J_{ij} \approx [(x_{i+1,j} - x_{i-1,j})(y_{i,j+1} - y_{i,j-1}) - (x_{i,j+1} - x_{i,j-1})(y_{i+1,j} - y_{i-1,j})]/4 \qquad (4.2.3)$$

or

$$Ag_{ij} = J_{ij} \approx [(x_{i+1/2,j} - x_{i-1/2,j})(y_{i,j+1/2} - y_{i,j-1/2}) - (x_{i,j+1/2} - x_{i,j-1/2})(y_{i+1/2,j} - y_{i-1/2,j})] \qquad (4.2.4)$$

The cell areas represented by these two equations are depicted in Fig. 4.3. In this investigation, Eq. (4.2.3) is used.

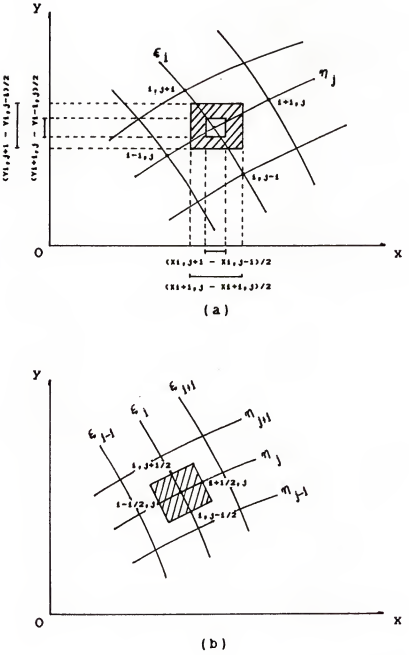


Figure 4.3 Correspondence between the Jacobian and the cell area.
(a) The shaded area is the cell area defined by Eq. (4.2.3)
(b) The shaded area is the cell area defined by Eq. (4.2.4)

With the concept of cell area defined, the algorithm for controlling smoothness can be described in the following steps:

(1) Set j=2.

(2) Determine J^{n}_{max} and J^{n}_{min} along the j^{th} grid line.

(3) Calculate ω^{n}_{max} and ω^{n}_{min} along the j^{th} grid line by using Eq. (3.2.14) where U is the solution at the n^{th} time level.

(4) Calculate ω^{n+1}_{max} and ω^{n+1}_{min} along the j^{th} grid line by using Eq. (3.2.14) where U is the tentative solution at $(n+1)^{th}$ time level.

(5) Calculate the ratio $(J_{max}/J_{min})^{n+1}$ by using Eq. (3.2.16) where $\Delta_{max}/\Delta_{min}$ are replaced by J_{max}/J_{min}.

(6) Calculate the smoothness factor SF by using Eq. (3.2.13) with Δ_{max} and Δ_{min} replaced by J^{n+1}_{max} and J^{n+1}_{min}, respectively.

(7) Replace j by j+1 and repeat steps 2 to 6 until j=JL-1.

(8) Repeat step 1 to 7 for the i^{th} grid line from i=2 to i= IL-1.

(9) Calculate the J^{n+1}_{ij}.

The grid system generated by the following steps 1-9 described above could contain overlapping grid lines. The sufficient condition for non-overlapping grid lines in the physical domain is that the cell area associated with each grid point (or the Jacobian) never vanishes. This can be achieved by controlling the value of J^{n+1}_{ij}. If J^{n+1}_{ij} calculated in step 9 is greater than J_{crt} (a value given a priori), then the algorithm can be stopped. If J^{th}_{ij} calculated in step 9 is less than J_{crt}, then the algorithm continues from step 9 with the following step:

(10) Rewrite Eq. (3.2.16) as follows:

$$\left|\frac{J_{max}}{J_{min}}\right|^{n+1} = \left|\frac{J_{max}}{J_{min}}\right|^{n} \frac{\left|\frac{\omega_{max}}{\omega_{min}}\right|^{n}}{\left|\frac{\omega_{max}}{\omega_{min}}\right|^{n+1}} \tag{4.2.5}$$

(11) Repeat steps 2-6 for all i^{th} and j^{th} grid lines where $J_{ij}<J_{crt}$ with one exception. The exception is in step 5 instead of using Eq. (3.2.16), use Eq. (4.2.5).

This completes the algorithm for controlling nonoverlapping of grid lines and smoothness. In the next section a technique for controlling the orthogonality of the grid system is presented.

4.3 Orthogonality of the Grid System

As noted earlier, grid systems for two-dimensional spatial domains should be nearly orthogonal. In Chapter I, it was explained that nearly orthogonal means that the grid lines should intercept each other with an angle approximately between 45 and 135 degrees. This fact can be used to control near orthogonality as shown in Fig 4.4.

In Fig. 4.4, the locations of grid points A and B have already been determined and the lines Aa, Ab, Bc and Bd define a sector of the physical domain in which the coordinates of the new grid point to be generated can be located to satisfy the minimum requirements for the grid to be nearly orthogonal.

The lines la and lb emanating from point A can be represented by the following equations:

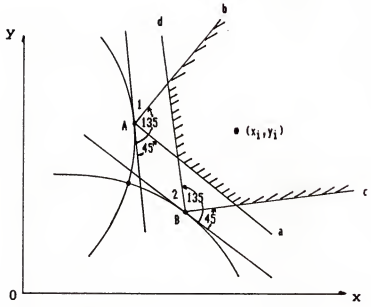


Figure 4.4 Orthogonality of grid lines in the physical domain.

Line 1a

$$y = m_{1a}\, x + h_{1a} \tag{4.3.1}$$

$$m_{1a} = \tan(\arctan m_1 + \pi/4) \tag{4.3.2}$$

$$h_{1a} = y_1 - m_{1a}\, x_1 \tag{4.3.3}$$

Line 1b

$$y = m_{1b}\, x + h_{1b} \tag{4.3.4}$$

$$m_{1b} = \tan(\arctan m_1 + 3\pi/4) \tag{4.3.5}$$

$$h_{1b} = y_1 - m_{1b}\, x_1 \tag{4.3.6}$$

where m_1 is the tangent to the grid line at grid point A.

Similarly, lines 2c and 2d emanating from grid point B can be represented by the following equations:

Line 2c

$$y = m_{2c}\, x + h_{2c} \tag{4.3.7}$$

$$m_{2c} = \tan(\arctan m_2 + \pi/4) \tag{4.3.8}$$

$$h_{2c} = y_2 - m_{2c}\, x_2 \tag{4.3.9}$$

Line 2d

$$y = m_{2d}\, x + h_{2d} \tag{4.3.10}$$

$$m_{2d} = \tan(\arctan m_2 + 3\pi/4) \tag{4.3.11}$$

$$h_{2d} = y_2 - m_{2d}\, x_2 \tag{4.3.12}$$

where m_2 is the tangent to the grid line at grid point B.

Finally, to accomplish the requirements of near orthogonality, the coordinates of the grid point to be generated have to satisfy the following inequalities:

$$y > m_{1a} x + h_{1a} \tag{4.3.13}$$

$$y > m_{2c} x + h_{2c} \tag{4.3.14}$$

$$x > (y - h_{1b})/m_{1b} \tag{4.3.15}$$

$$x > (y - h_{2d})/m_{2d} \tag{4.3.16}$$

The inequalities expressed by Eqs. (4.3.13) to (4.3.16) were used to locate the coordinates of the grid points in the physical domain to ensure the grid generated would be nearly orthogonal during the application of the two-dimensional grid generation method developed.

This concludes the description of the solution-adaptive grid generation method for two-dimensional, rectangularly shaped spatial domains. Several examples illustrating how the method developed in this chapter can be applied for generating two-dimensional grid systems are presented in Chapter VIII.

CHAPTER V

THE SAAGG METHOD FOR 2-D, ARBITRARILY SHAPED DOMAINS

The two-dimensional, solution-adaptive, algebraic grid generation (SAAGG) method developed in Chapter IV can be used with any non-adaptive, grid generation technique to generate solution-adaptive grid systems in arbitrarily shaped 2-D spatial domains that can deform in an arbitrary manner. This is demonstrated in this investigation by using the method developed in Chapter IV in conjunction with the Two-Boundary Technique. This demonstration is presented in the following order. First, the Two-Boundary Technique is briefly described. Afterwards, the criteria for controlling the nonoverlapping of grid lines in the physical domain and the orthogonality of the grid system at the boundaries are discussed. Finally, how the method developed in Chapter IV is incorporated into the Two-Boundary Technique is described.

5.1 Overview of the Two-Boundary Technique

The Two-Boundary Technique is a non-solution-adaptive, algebraic grid generation method based on interpolation techniques [26]. In general, Hermite interpolation is applied between two boundaries of the spatial domain to establish the coordinate transformation which maps the physical domain onto the computational domain [27]. Here, it is important to note that the Two-Boundary Technique only maps two boundaries of

the spatial domain correctly. The other two boundaries of the spatial domain can be mapped correctly only if they are straight lines [3,18]. In practice, there are some problems in which it is necessary to map four boundaries of the spatial domain correctly. In these cases, the Four-Boundary Technique can be used instead of the Two-Boundary Technique [3].

The Two-Boundary Technique can be explained by considering Fig. (5.1) where the correspondence between the boundaries of the physical and the computational domains is shown. In that figure, the two boundaries AD and BC of the spatial domain are described by the following parametric equations:

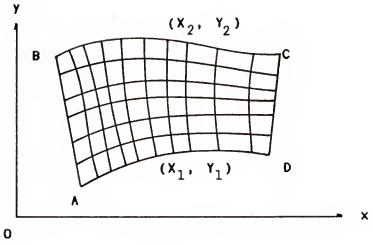
$$x_1 = x_1(\xi,\eta{=}0,\tau) \qquad y_1 = y_1(\xi,\eta{=}0,\tau) \tag{5.1.1,2}$$

$$x_2 = x_2(\xi,\eta{=}1,\tau) \qquad y_2 = y_2(\xi,\eta{=}1,\tau) \tag{5.1.3,4}$$

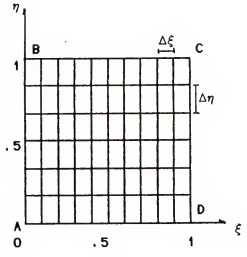
where the subscripts 1 and 2 denote boundaries AD and BC, respectively. Note that since x_1, x_2, y_1, y_2 are functions of τ, the boundaries AD and BC can deform with time.

The Two-Boundary Technique based on Hermite interpolation gives the following coordinate transformation between the physical domain in x-y-t coordinate system and the computational domains in ξ-η-τ coordinate system [3,28]:

$$x(\xi,\eta,\tau) = x_1(\xi,\tau)h_1(\eta) + x_2(\xi,\tau)h_2(\eta) + \frac{\partial x(\xi,\eta{=}0,\tau)}{\partial \eta}h_3(\eta) + \frac{\partial x(\xi,\eta{=}1,\tau)}{\partial \eta}h_4(\eta) \tag{5.1.5}$$



(a)



(b)

Figure 5.1 Correspondence between the boundaries of the physical domain and the computational domain.
(a) Arbitrarily shaped physical domain (x-y-t).
(b) Computational domain (ξ-η-τ).

$$y(\xi,\eta,\tau) = y_1(\xi,\tau)h_1(\eta) + y_2(\xi,\tau)h_2(\eta) +$$

$$\frac{\partial y(\xi,\eta=0,\tau)}{\partial \eta} h_3(\eta) + \frac{\partial y(\xi,\eta=1,\tau)}{\partial \eta} h_4(\eta) \qquad (5.1.6)$$

The coefficients h_1, h_2, h_3, and h_4 in the equations above are given by the following expressions [3]:

$$h_1(\eta) = 2\eta^3 - 3\eta^2 + 1 \qquad (5.1.7)$$

$$h_2(\eta) = -2\eta^3 + 3\eta^2 \qquad (5.1.8)$$

$$h_3(\eta) = \eta^3 - 2\eta^2 + \eta \qquad (5.1.9)$$

$$h_4(\eta) = \eta^3 - \eta^2 \qquad (5.1.10)$$

The partial derivative terms appearing in Eqs. (5.1.5) and (5.1.6) are chosen so that grid lines intersect the boundaries AD and BC perpendicularly in the physical domain. Accordingly, these partial derivative terms are given as follows [3]:

$$\frac{\partial x(\xi,\eta=0,\tau)}{\partial \eta} = - K_1(\xi) \frac{\partial y_1}{\partial \xi} \qquad (5.1.11)$$

$$\frac{\partial y(\xi,\eta=0,\tau)}{\partial \eta} = + K_1(\xi) \frac{\partial x_1}{\partial \xi} \qquad (5.1.12)$$

$$\frac{\partial x(\xi,\eta-1,\tau)}{\partial \eta} = - K_2(\xi) \frac{\partial y_2}{\partial \xi} \tag{5.1.13}$$

$$\frac{\partial y(\xi,\eta-1,\tau)}{\partial \eta} = + K_2(\xi) \frac{\partial x_2}{\partial \xi} \tag{5.1.14}$$

The coefficients K_1 and K_2 appearing in the equations above are determined by trial and error to ensure nonoverlapping of the grid lines in the physical domain. In order to incorporate the solution-adaptive method developed in Chapter IV into the Two-Boundary Technique, these coefficients must be determined analytically.

5.2 Extended Two-Boundary Technique

In this section, the Two-Boundary Technique is extended so that it can be applied with the method developed in Chapter IV. First, a procedure is described for expressing the coefficients K_1 and K_2 in analytical form. Second, criteria are defined for ensuring that grid lines do not overlap each other or the boundaries. Third, a technique is presented which ensures that the angles at which grid lines intersect the two boundaries are within certain limits.

5.2.1. Analytical Expressions for K1 and K2

The presentation of the procedure for determining the coefficients K_1 and K_2 is facilitated by substituting Eqs. (5.1.11) to (5.1.14) into Eqs. (5.1.5) and (5.1.6) as shown below.

$$x(\xi,\eta,\tau) = x_1h_1 + x_2h_2 - K_1B_1h3 - K_2B_2h_4 \qquad (5.2.1)$$

$$y(\xi,\eta,\tau) = y_1h_1 + y_2h_2 + K_1A_1h_3 + K_2A_2h_4 \qquad (5.2.2)$$

where the partial derivatives appearing in Eqs. (5.1.11) to (5.1.14) were represented by

$$A_1 = \frac{\partial x_1}{\partial \xi} \qquad A_2 = \frac{\partial x_2}{\partial \xi} \qquad (5.2.3,4)$$

$$B_1 = \frac{\partial y_1}{\partial \xi} \qquad B_2 = \frac{\partial y_2}{\partial \xi} \qquad (5.2.5,6)$$

In order to determine analytical expressions for K_1 and K_2, first, a grid line should be specified, a priori, inside the physical domain. In this investigation, the locations of grid points along this grid line were chosen to be

$$x(\xi,\eta{=}0.5) = Q(\xi) = \epsilon + \frac{x_1 + x_2}{2} \qquad (5.2.7)$$

$$y(\xi,\eta{=}0.5) = P(\xi) = y_1 + \frac{y_2 + y_1}{2} \qquad (5.2.8)$$

where x_1, x_2, y_1, and y_2 are functions of ξ and are given by Eqs. (5.1.1) to (5.1.4), and ϵ is a small quantity between $\pm\ 0.1\Delta x$.

Next, the values of h_1, h_2, h_3, and h_4 needed to be determined. Since η=0.5, Eqs. (5.1.7) to (5.1.10) give

$$h_1 = h_2 = \frac{1}{8} \quad \text{and} \quad h_3 = -h_4 = \frac{1}{8} \qquad (5.2.9,10)$$

Substituting Eqs. (5.2.7) to (5.2.10) into Eqs. (5.2.1) and (5.2.2), and solving for the coefficients K_1 and K_2 yield the following expressions:

$$K_1 = \frac{8(QA_2 + PB_2) - 4[A_2(x_1 + x_2) + B_2(y_1 + y_2)]}{(A_1B_2 - A_2B_1)} \qquad (5.2.11)$$

$$K_2 = \frac{8(QA_1 + PB_1) - 4[A_1(x_1 + x_2) + B_1(y_1 + y_2)]}{(A_1B_2 - A_2B_1)} \qquad (5.2.12)$$

The equations above are the desired analytical expressions for the coefficients K_1 and K_2.

Here, it is noted that the expressions for the coefficients K_1 and K_2 given by Eqs. (5.2.11) and (5.2.12) approach infinity when the denominator in those equations approaches zero. This problem can be overcome by setting K_1 and K_2 equal to the average values of the coefficients immediately before and after the grid point in which Eqs. (5.2.11) and (5.2.12) become singular.

The analytical expressions for K_1 and K_2 defined by Eqs. (5.2.11) and (5.2.12) will be useful when the solution-adaptive grid generation method developed in Chapter IV is used with the Two-Boundary Technique. The influence of the coefficients K_1 and K_2 in controlling the distribution of the grid points in the physical domain is studied in the next section. This study is made to establish the criteria for nonoverlapping grid lines.

5.2.2 Criteria for Nonoverlapping Grid Lines

The grid η-grid lines in the physical domain are determined by using Eqs. (5.2.1) and (5.2.2) and setting η to a constant between 0 and 1 in Eqs. (5.1.7) to (5.1.10). The values of K_1 and K_2 determine whether any given η-grid line will be located closer or farther away from the boundaries. Thus, it is important to determine the limits of these coefficients in order to control the nonoverlapping of grid lines.

The η-grid lines will not overlap the two boundaries AD and BC of the physical domain if the y-coordinates of those grid lines satisfy the following inequalities:

$$y(\xi,\Delta\eta) > y_1(\xi,0) \tag{5.2.13}$$

$$y(\xi,1-\Delta\eta) < y_2(\xi,1) \tag{5.2.14}$$

By using Eqs. (5.1.6) to (5.1.9) and setting $\eta=1-\Delta\eta$, the following equations can be written:

$$h_1 = 2(1-\Delta\eta)^3 - 3(1-\Delta\eta)^2 + 1 \tag{5.2.15}$$

$$h_2 = -2(1-\Delta\eta)^3 + 3(1-\Delta\eta)^2 \tag{5.2.16}$$

$$h_3 = (1-\Delta\eta)^3 - 2(1-\Delta\eta)^2 + (1-\Delta\eta) \tag{5.2.17}$$

$$h_4 = (1-\Delta\eta)^3 - (1-\Delta\eta)^2 \tag{5.2.18}$$

By neglecting the higher order terms in $\Delta\eta$, the equations above give $h_1=0$, $h_2=1$, $h_3=0$, and $h_4=-\Delta\eta$. By substituting these results into Eq. (5.2.2) and the resultant equation into Eq. (5.2.14), the following inequality can be written:

$$K_2 \, \Delta\eta \, A_2 > 0 \tag{5.2.15}$$

Since $\Delta\eta$ is always positive and the partial derivative A_2 is also positive, the only solution which satisfies the inequality above is

$$K_2 > 0 \tag{5.2.16}$$

Similarly, for $\eta=\Delta\eta$ the following inequality can be found:

$$K_1 > 0 \tag{5.2.17}$$

Thus, the criterion for the grid lines not to overlap the boundaries AD and BC of the physical domain is that the coefficients K_1 and K_2 are always positive.

The criteria for nonoverlapping η-grid lines in the interior of the physical domain are very involved and are described below for spatial domains which contains a line of symmetry. For those spatial domains, it can be deduced from Eqs. (5.2.1) to (5.2.4) that $A_1=A_2=A$, and $B_1=-B_2=B$. Similarly from Eqs. (5.2.11) and (5.2.12), it can be deduced that $K_1=K_2=K$.

To establish the criteria for nonoverlapping of interior η-grid lines, it is necessary to investigate the influence of the coefficient K on the distribution of grid points. This influence was analyzed, in this investigation, through the ratio between Δm_i and Δb_i determined by the coordinates of grid points at r, s, t and m, along the grid line ξ_i, shown in Fig. 5.2. This ratio is given by

$$\alpha_i = \frac{\Delta m_i}{\Delta b_i} \tag{5.2.18}$$

where

$$\Delta m_i = \pm [(x_m - x_t)^2 + (y_m - y_t)^2]_i^{1/2} \tag{5.2.19}$$

and

$$\Delta b_i = \pm [(x_s - x_r)^2 + (y_s - y_r)^2]_i^{1/2} \tag{5.2.20}$$

In the above equations, the negative sign is used to indicate those cases in which $(y_m - y_t) < 0$ or $(y_s - y_r) < 0$. The coordinates of the grid points at s, t and m along grid line ξ_i are given by Eqs. (5.2.1) and (5.2.2) by setting $\eta=\Delta\eta$, $\eta=0.5-\Delta\eta$, and $\eta=0.5$, respectively. The coordinates of the grid point at r along the boundary AD are given

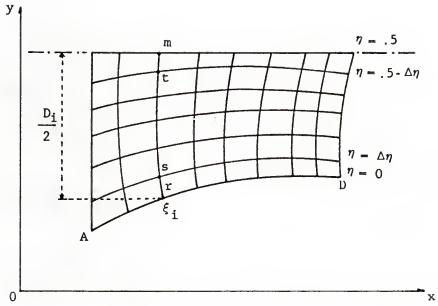


Figure 5.2 Grid lines in a spatial domain which contains a line of symmetry.

by Eqs. (5.1.1) and (5.1.2). The substitution of the coordinates thus obtained into Eqs. (5.2.19) and (5.2.20) gives an expression for Δm_i and Δb_i as function of ξ_i and K_i. Hence, the ratio α_i can be expressed in terms of ξ_i and K_i as follows:

$$\alpha_i = \alpha(\xi_i, K_i) \tag{5.2.21}$$

The exact functional form of Eq. (5.2.21) is not shown because it depends on the parametric equation used to describe the boundary AD of the spatial domain. Figure 5.3 shows a typical relationship between α and K for several grid lines ξ_i. In Fig. 5.3, also three regions can be distinguished. When $\alpha > 1$, η-grid lines tend to cluster near the boundaries. When $0 < \alpha < 1$, η-grid lines tend to stretch away from the boundaries. Finally, when $\alpha < 0$, η-grid lines overlap.

A few other observations can also be made from Fig. 5.3. First, if $K < Ka$, then every grid point along η-grid lines in the physical domain is clustered to the boundaries. Second, if $Ka < K < Kb$, then some grid points along η-grid lines will be clustered near the boundary while others will be stretched away from the boundary depending upon the value of α_i. Third, if $Kb < K < Kc$, then every grid point along η-grid line will be stretched away from the boundary. Finally, if $K > Kc$, then some η-grid lines will be overlaped.

The limits Ka, Kb, and Kc should be determined before the solution-adaptive grid generation method developed in Chapter IV is to be applied with the Two-Boundary Technique. This is because the locations of the grid points in physical domain are influenced by the coefficients K as

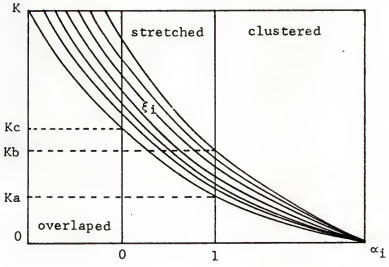


Figure 5.3 Relationship between the ratio $\alpha_i = \Delta m_i / \Delta b_i$ and the coefficient K.

well as by the stretching function defined in Chapter II.

Thus, the criterion for nonoverlapping grid lines is that the ratio α_i be positive for all of the grid lines in the physical domain; i.e.,

$$\alpha_i = \frac{\Delta m_i}{\Delta b_i} > 0 \tag{5.2.22}$$

It was shown in this section that the coefficient K must be positive for the grid lines to not overlap the two boundaries AD and BC of the physical domain. Thus, if K is positive, then Δb_i must be positive which implies from Eq. (5.2.22), that Δm_i must be also positive.

From Eq. (5.2.19), it can be seen that the condition for Δm_i to be positive is that

$$y_m - y_t > 0 \tag{5.2.23}$$

Since η=0.5 is a line of symmetry, y_m in the equation above is determined from the following expression:

$$y_m = y_1 + \frac{D}{2} \tag{5.2.24}$$

where y_1 is obtained from Eq. (5.1.2) and D/2 is the distance shown in Fig. 5.2. In Eq. (5.2.23), y_t is obtained from Eq. (5.2.2). In this equation the coefficients h_1, h_2, h_3 and h_4 are determined from Eqs. (5.1.7) to (5.1.10) by setting η=0.5-$\Delta\eta$ and neglecting higher order terms in $\Delta\eta$. After simplification, the following equations are found:

$$h_1 = \frac{3}{2}\Delta\eta + \frac{1}{2} \tag{5.2.25}$$

$$h_2 = -\frac{3}{2}\Delta\eta + \frac{1}{2} \tag{5.2.26}$$

$$h_3 = \frac{1}{8} + \frac{\Delta\eta}{4} \tag{5.2.27}$$

$$h_4 = -\frac{1}{8} + \frac{\Delta\eta}{4} \tag{5.2.28}$$

The substitution of Eqs. (5.2.25) to (5.2.28) into Eq. (5.2.2) gives

$$y_t = y_1\left(\frac{3}{2}\Delta\eta + \frac{1}{2}\right) + (y_1 + D)\left(\frac{1}{2} - \frac{3}{2}\Delta\eta\right) + (KA)\frac{\Delta\eta}{2}$$

$$y_t = y_1 + \frac{D}{2} - \frac{\Delta\eta}{2}(3D - KA) \tag{5.2.29}$$

Finally, the substitution of Eqs. (5.2.24) and (5.2.29) into Eq. (5.2.23) gives the following expression:

$$\frac{\Delta\eta}{2}(3D-KA) > 0 \tag{5.2.30}$$

Since $\Delta\eta$ is positive and K is also positive, the criterion for non-overlapping η-grid lines in the physical domain becomes

$$0 < K < \frac{3D}{A} \tag{5.2.31}$$

Here, it is noted that the inequality $K > 0$ is valid for all types of spatial domains but the second inequality $K < 3D/A$ is only valid for those spatial domains which possess one line of symmetry between two boundaries AD and BC.

5.2.3 Criteria for Orthogonality at Boundaries

By using the Two-Boundary Technique with Hermite interpolation, Eqs. (5.1.11) to (5.1.14) ensure that the grid lines intersect orthogonally the boundaries of the physical domain. However, this is not true after the continuous grid lines have been replaced by a finite number of grid points. In this section, the Two-Boundary Technique is extended to ensure that the deviation from orthogonality at the boundaries of the physical domain is within certain limits.

The deviation from orthogonality at the boundaries (θ) is shown in Fig. 5.4. In this figure it can be seen that the angle θ is formed by the vector normal to the boundary $\vec{e}_\xi$ and the vector $\vec{a}$.

The vector normal to the boundary can be expressed by

$$\vec{e}_\xi = \frac{\partial x(\xi_i,\eta=0)}{\partial \xi}\,\vec{i} + \frac{\partial y(\xi_i,\eta=0)}{\partial \xi}\,\vec{j} \tag{5.2.32}$$

where the x and y are given by Eqs. (5.2.1) and (5.2.2), respectively.

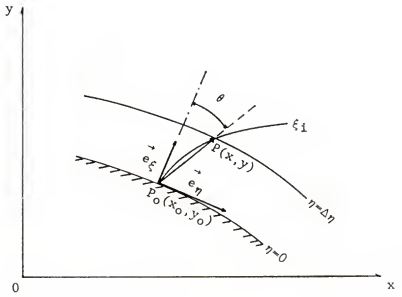


Figure 5.4 Deviation from orthogonality at the boundaries of the physical domain.

The vector $\vec{a}$ is expressed in terms of the coordinates of grid points $P(x,y)$ and $P_o(x_o,y_o)$ shown in Fig. 5.4, as follows:

$$\vec{a} = (x - x_o)\,\vec{i} + (y - y_o)\,\vec{j} \tag{5.2.33}$$

The coordinates of grid point $P(x,y)$ are determined by Eqs. (5.2.1) and (5.2.2) for a given grid line ξ_i and with $\eta=\Delta\eta$ as function of the coefficients $K_1(\xi_i)$ and $K_2(\xi_i)$.

The coordinates of grid point $P_o(x_o,y_o)$ at the boundary of the physical domain are determined by Eqs. (5.1.1) and (5.1.2) once the grid line ξ_i has been specified.

The angle θ between the vectors $\vec{e}_\xi$ and $\vec{a}$ is obtained from analytical geometry and is given by

$$\theta = \cos^{-1} \frac{\vec{e}_\xi \cdot \vec{a}}{|\vec{e}_\xi|\ |\vec{a}|} \tag{5.2.34}$$

By substituting Eqs. (5.2.32) and (5.2.33) into the equation above and effectuating the vectorial operation, the following equation is obtained:

$$\theta = \cos^{-1} \frac{E_1C_1 + F_1D_1}{[(E_1^2+F_1^2)(C_1^2+D_1^2)]^{1/2}} \tag{5.2.35}$$

where the following substitutions were made:

$$E_1 = \frac{\partial x(\xi_i,\eta=0)}{\partial \xi} \qquad F_1 = \frac{\partial y(\xi_i,\eta=0)}{\partial \xi} \qquad (5.2.36,37)$$

$$C_1 = (x - x_o) \qquad D_1 = (y - y_o) \qquad (5.2.38,39)$$

Equation (5.2.35) is a function of ξ_i, $\Delta\eta$, and the coefficients K_1 and K_2; i.e.,

$$\theta_i = \theta(\xi_i,\Delta\eta,K_1,K_2) \qquad (5.2.40)$$

The exact functional form of Eq. (5.2.40) is not shown because it depends on the parametric equation used to describe the boundaries of the spatial domain. Figure 5.5 shows a typical relationship between θ and ξ_i for a given value of $\eta=\Delta\eta$ and several values of K.

Figure 5.5 shows that the bigger the coefficient K the smaller is the deviation from orthogonality at the boundaries. Unfortunately, constraints imposed by Eq. (5.2.31), to ensure the grid lines do not overlap, sometimes, do not allow the coefficient K to be large enough to keep the deviation from orthogonality within the limits required for implementation of the boundary conditions.

In this investigation, two techniques were developed for controlling the orthogonality at the boundaries. The first technique controls orthogonality by specifying the minimum number of grid points JL, in y-direction, needed to keep the deviation from orthogonality within a prescribed limit. The second technique controls the orthogonality by determining a stretching function which redistributes existing grid points to keep the deviation within the prescribed limit.

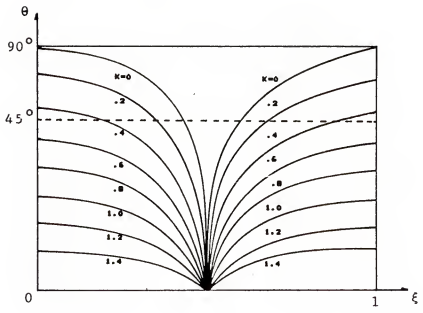


Figure 5.5 Deviation from orthogonality at the boundaries of the physical domain as function of ξ_i and the coefficient K.

Figure 5.6 shows how the deviation from orthogonality can be reduced by clustering the grid lines to the boundaries of the physical domain. In this figure, the angle θ is the deviation from orthogonality at the boundary before stretching and β is the maximum deviation from orthogonality allowed.

To reduce the deviation from orthogonality by the first technique, Eq. (5.2.35) is rewritten to introduce the maximum deviation from orthogonality allowed, as shown below:

$$\beta = \cos^{-1} \frac{E_1 C_1 + F_1 D_1}{[(E_1^2 + F_1^2)(C_1^2 + D_1^2)]^{1/2}} \qquad (5.2.41)$$

Once the maximum coefficient K has been determined by Eq. (5.2.31) everything on the right hand side of Eq. (5.2.40) is known except for the variables C_1 and D_1. These variables are defined by Eqs. (5.2.36) and (5.2.37) which are expressed as functions of the ξ_i and $\Delta\eta$. Thus, for the maximum angle of deviation β and a given grid line ξ_i, a new value of $\Delta\eta$, denoted by $\Delta\eta^*$, can determined from Eq. (5.2.41).

The grid spacings $\Delta\eta$ and $\Delta\eta^*$ are related to the number of grid lines by the following expressions:

$$\Delta\eta = \frac{1}{JL - 1} \qquad \Delta\eta^* = \frac{1}{JL^* - 1} \qquad (5.2.42,43)$$

Now, the minimum number of grid points which will keep the deviation from orthogonality no greater than β is determined by substituting the

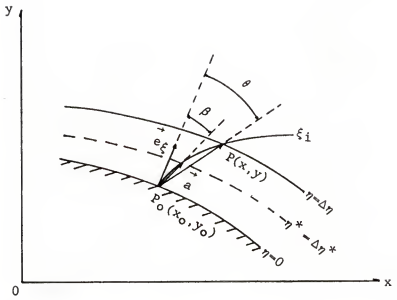


Figure 5.6 Reduction of deviation from orthogonality by using stretching function.

value of $\Delta\eta^*$ obtained from Eq. (5.3.41) in Eq. (5.3.43), as follows:

$$JL^* = 1 + \frac{1}{\Delta\eta^*} \tag{5.2.44}$$

The second technique to reduce the deviation from orthogonality was developed for problems in which the number of grid points cannot be changed. For these problems, a stretching function must be determined in order to redistribute the grid points in the physical domain and move the grid lines toward the boundaries.

To demonstrate the second technique, let the stretching function be a second-order polynomial shown below:

$$\eta^* = A\eta^2 + B\eta + C \tag{5.2.45}$$

where η and η^* are the η-coordinates before and after stretching. The coefficients A, B, and C in Eq. (5.2.45) are defined by:

$$B = \frac{(JL - 1)^2 - 2(JL^* - 1)}{(G - 3)(JL^* - 1)} \tag{5.2.46}$$

$$A = 2(1 - B) \tag{5.2.47}$$

$$C = 0 \tag{5.2.48}$$

when $\eta \leq 0.5$ and

$$B = \frac{2(G^2 - 1) - 3(G^* - 1)}{(G - 1)(2G - 1)} \tag{5.2.49}$$

$$A = \frac{2}{3}(1 - B) \tag{5.2.50}$$

$$C = \frac{1}{3}(1 - B) \tag{5.2.51}$$

when $\eta > 0.5$. In Eqs. (5.2.46) through (5.2.51) the following substitutions were made:

$$G = \frac{JL - 2}{JL - 1} \quad \text{and} \quad G^{*} = \frac{JL^{*} - 2}{JL^{*} - 1} \tag{5.2.52,53}$$

The stretching function defined by Eq. (5.2.45) redistribute the grid points in the physical domain so that deviation from orthogonality was no greater than β.

This concludes the description of the extended Two-Boundary Technique. In the next section, an algorithm for applying the solution-adaptive grid generation method developed in Chapter IV with the extended Two-Boundary Technique is presented.

5.3 Algorithm for 2-D, Solution-Adaptive Grid Systems

This section presents the algorithm for generating solution-adaptive grid systems for two-dimensional problems in which the boundaries of the physical domain can move or be stationary. The algorithm will map the grid points between the physical domain in x-y-t coordinates system and the computational domain in ξ-η-τ coordinates system by using the SAAGG method present in Chapter IV in conjunction with the Two-Boundary

Technique. The algorithm is as follows:

(1) Transform the governing equations from the coordinate system of the physical domain to the coordinate system of the computational domain.

(2) Apply any suitable finite-difference method to express the transformed governing equations as a set of finite-difference equations (FDEs).

(3) Specify the number of grid lines IL in the ξ-direction and the number of grid line JL in the η-direction.

(4) Set the initial time level, n, to zero (which corresponds to time t=0) and specify the duration of interest T (i.e. $0 \leq t \leq T$).

(5) Decide the correspondence between the boundaries of the physical domain and the boundaries of the computational domain.

(6) Express the coordinates of two boundaries of physical domain in terms of the boundary-fitted coordinates as shown by Eqs.(5.1.1) to (5.1.4).

(7) Determine the partial derivatives appearing in Eqs. (5.2.1) and (5.2.2) by using expressions obtained in step 6.

(8) Specify the locations of the grid points in the computational domain.

$$\xi_i = \frac{i - 1}{IL - 1} \qquad i=1,2,\ldots,IL \qquad (5.2.54)$$

$$\eta_i = \frac{j - 1}{JL - 1} \qquad j=1,2,\ldots,JL \qquad (5.2.55)$$

(9) Calculate the coefficients h_1, h_2, h_3 and h_4 by using Eqs. (5.1.7) to (5.1.10).

(10) Calculate the coordinates of grid points given by the functions P and Q shown in Eqs. (5.2.7) and (5.2.8).

(11) Determine the coefficient K by using P and Q calculated in step 10 and Eqs.(5.2.11) and (5.2.12).

(12) Use the coefficients calculated in step 11 along with the criteria expressed by Eq. (5.2.31) to verify nonoverlapping of grid lines.

(13) Determine the locations of the grid points in the physical domain at the initial time level (n=0) by using Eqs. (5.2.1) and (5.2.2) with the parameters evaluated in the previous steps.

(14) Determine the coefficients Ka, Kb, and Kc by using Eq. (5.2.21) with the grid system obtained in step 13. If K < Ka, then grid lines cluster near the boundaries. If Kc > K > Kb, then grid lines stretching away from the boundaries.

(15) Specify the maximum deviation from orthogonality permitted at boundaries (β).

(16) Compare the deviation from orthogonality θ determined from Eq. (5.2.40) with β given in step 15. If $\theta \leq \beta$ then continue the algorithm from step 19 otherwise continue from step 17.

(17) Determine the value of coordinate $\Delta\eta^*$ by using Eq. (5.2.41).

(18) Determine the new number of grid points JL^* by using Eq.(5.2.44) and repeat the algorithm from step 3.

(19) At this point, use the same procedure described by steps 8 to 14 in the algorithm for 2-D, rectangularly shaped spatial domains

presented in Chapter IV. Here, it is noted that the gradient used to determine the weighting function in Eq. (3.2.5) for the x-direction, may be different from the one used for the y-direction.

(20) Replace the time level n by n+1 and calculate the total time elapsed by using Eq. (3.3.11). Repeat steps 11 to 19 until the total time elapsed equals to the duration of interest specified in step 5.

This completes the description of the solution-adaptive, grid generation method for two-dimensional, arbitrarily shaped spatial domains in which the boundaries can move or be stationary.

CHAPTER VI

APPLICATIONS OF STRETCHING FUNCTIONS

In this chapter, the technique developed in Chapter III is used to formulate stretching functions for distributing grid points in one-dimensional spatial domains. This chapter is divided in three sections. In the first section, a stretching function is formulated to be used with steady-state problems in which the solution is known qualitatively but not quantitatively. This stretching function did not have to be coupled to the solution because the regions where grid points were most needed were known a priori. In the second section, stretching functions were formulated for unsteady problems in which the solution at some specified time is known. In these cases, it is shown that the stretching function should be coupled to the solution to determine the locations of grid points as the problem is being computed. Finally, in the third section, the method developed was used to generate grid points necessary to solve a one-dimensional initial-value problem using finite-difference methods. The solution obtained in this section by using the SAAGG technique was compared to the known analytical solution, as well as to the solution obtained by using equally spaced grid points.

6.1 Stretching Functions Not Coupled to the Solution

The technique developed in Chapter III was designed for formulating stretching functions which are coupled to the solution. However, the

same technique also can be used for formulating stretching functions which are not coupled to the solution. In order to formulate such non-coupled stretching functions, it is necessary to specify the weighting function appearing in Eq. (3.1.25) based on the qualitative interpretation of the physics of the problem. An example of a non-coupled stretching function is formulated in this section.

Suppose that the qualitative interpretation of the problem being analized suggests that grid points be clustered toward one boundary of the spatial domain. As noted earlier, grid points will be clustered in regions where the weight function is high. Here, the weighting function was chosen to be the inverse of a parabola whose vertex is in the vicinity of the boundary where the grid points should be clustered. Figure 6.1 shows this weighting function and the boundaries of the spatial domain which are at L_1=1 and L_2=L.

The mathematical expression for that weighting function is

$$W(x) = \frac{C_2}{Ax^2 + Bx + C} \tag{6.1.1}$$

where the constant C_2 is defined by Eq. (3.1.24) and the constants A, B and C are arbitrary constants. In this problem, the x-coordinate of the vertex of the parabola appearing in the denominator of Eq. (6.1.1) was set at x=1/2. Thus, the relationship between A and B can be determined by setting the derivative of the denominator in Eq. (6.1.1) at x=1/2 to zero, yielding

$$B = -A \tag{6.1.2}$$

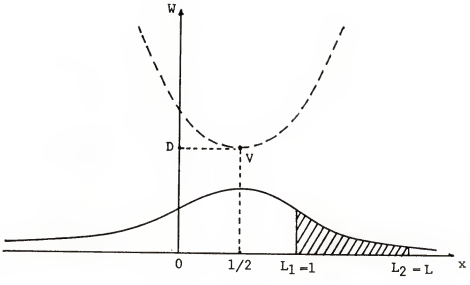


Figure 6.1 Weighting function for generating stretching functions to cluster grid points toward one boundary of the spatial domain.

By substituting Eq.(6.1.2) into Eq.(6.1.1), the following weighting function is obtained:

$$W(x) = \frac{C_2}{Ax^2 - Ax + C} \tag{6.1.3}$$

With the above weighting function, Eq. (3.1.25) gives the following stretching function:

$$K(x) = \frac{1 + \int_1^x (Ax^2 - Ax + C)dx}{x} \tag{6.1.4}$$

Integrating the above equation gives the locations of the grid points after stretching, as follows:

$$\bar{x} = \left(\frac{Ax^3}{3} - \frac{Ax^2}{2} + Cx \right) - \left(\frac{A}{3} - \frac{A}{2} + C - 1 \right) \tag{6.1.5}$$

where $\bar{x}$ is defined by Eq. (3.1.4). Since A and C are arbitrary constants, the equation above can be simplified by setting the second parenthesis in Eq. (6.1.5) to zero. This gives the following relationship between A and C:

$$\frac{A}{3} - \frac{A}{2} + C - 1 = 0$$

$$C = 1 + \frac{A}{6} \tag{6.1.6}$$

By substituting Eq.(6.1.6) into Eq.(6.1.5) and factoring, the following equation is obtained:

$$\bar{x} = x \left[1 - \frac{A}{3}(x - 0.5)(1 - x) \right] \qquad (6.1.7)$$

By comparing the above equation with other stretching functions for the same purpose appearing in the literature, it was found that Eq. (6.1.7) is identical to the stretching function proposed by Vinokur [24] who developed a method for formulating stretching functions based on truncation error analysis. The Vinokur's stretching function designed to cluster grid points toward one boundary of the spatial domain is written below for comparison.

$$\bar{x} = x \left[1 - 2(1-\beta)(x - 0.5)(1 - x) \right] \qquad (6.1.8)$$

where β is a constant. By comparing the above equation with Eq.(6.1.7) obtained by using the technique developed in this investigation, it can be seen that the term $2(1-\beta)$ appearing in Eq. (6.1.8) is equivalent to the term $A/3$ appearing in Eq. (6.1.7).

6.2 Stretching Functions Coupled to the Solution

In this section, three examples are given to demonstrate the capability of the method developed for formulating stretching functions that are coupled to the solution as the problem is being computed.

6.2.1 Steep Gradients in One Side of the Spatial Domain

Suppose that the gradient of the solution for the problem being analyzed is known at some time level and is given by

$$\frac{du}{dx} = 0.075(x - 1)^{-1.075} \tag{6.2.1}$$

Also, suppose that the number of grid points to be used for solving this problem is fixed at IL=21 and that the boundaries of the spatial domain are at L_1=1 and L_2=21.

By substituting Eq. (6.2.1) into Eq. (3.2.5) gives the weighting function which is

$$W(x) = [\ 1 + |0.075(x - 1)^{-1.075}|\ ]^{SF} \tag{6.2.2}$$

With the above weighting function, the stretching function for this problem is given by Eq. (3.1.25); i.e.,

$$K(x) = \frac{1 + 20\ \dfrac{\displaystyle\int_1^x [1+|0.075(x - 1)^{-1.075}|]^{-SF} dx}{\displaystyle\int_1^{21} [1+|0.075(x - 1)^{-1.075}|]^{-SF} dx}}{x} \tag{6.2.3}$$

where SF is determined by Eq. (3.2.13).

The equation above is solved numerically and the grid system thus generated is shown in Fig. 6.2. In that figure it can be seen that the

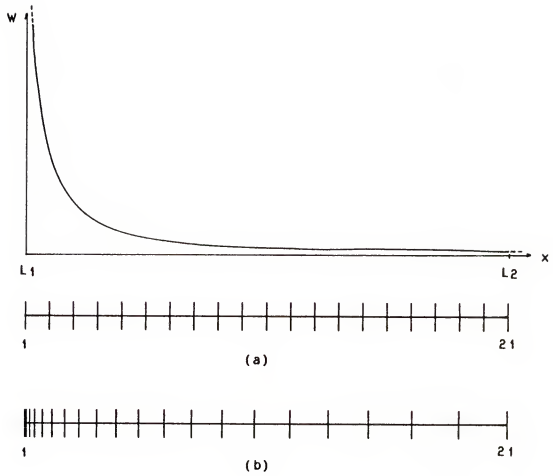


Figure 6.2 Grid system generated for problems with steep gradients in one side of the spatial domain.
(a) Grid system before stretching.
(b) Grid system after stretching.

grid points are clustered near the boundary where the gradient of the solution is high.

6.2.2 Steep Gradients in the Middle of the Spatial Domain

Suppose that the gradient of the solution for the problem being analyzed is known at some time level and is given by the following equations:

$$\frac{du}{dx} = 0.07(11 - x)^{-0.93} \qquad 1 \le x \le 11 \qquad (6.2.4)$$

$$\frac{du}{dx} = 0.07(x - 11)^{-0.93} \qquad 11 < x \le 21 \qquad (6.2.5)$$

Also suppose that the number of grid points to be used for solving this problem is fixed at IL=21 and the boundaries of the spatial domain are at L_1=1 and L_2=21.

By substituting Eqs. (6.2.4) and (6.2.5) into Eq. (3.2.5) gives the weighting functions which are

$$W(x) = [\ 1 + |0.07(11 - x)^{-0.93}|\ ]^{SF} \qquad 1 \le x \le 11 \qquad (6.2.6)$$

$$W(x) = [\ 1 + |0.07(x - 11)^{-0.93}|\ ]^{SF} \qquad 11 < x \le 21 \qquad (6.2.7)$$

With the above weighting functions, the stretching function for this

problem is given by Eq. (3.1.25), i.e, for $1 \le x \le 11$

$$K(x) = \frac{1 + 10 \dfrac{\displaystyle\int_1^x [1+|0.07(x - 11)^{-0.93}|]^{-SF} dx}{\displaystyle\int_1^{11} [1+|0.07(x - 11)^{-0.93}|]^{-SF} dx}}{x} \qquad (6.2.8)$$

and for $11 \le x \le 21$

$$K(x) = \frac{11 + 10 \dfrac{\displaystyle\int_{11}^x [1+|0.07(x - 11)^{-0.93}|]^{-SF} dx}{\displaystyle\int_{11}^{21} [1+|0.07(x - 11)^{-0.93}|]^{-SF} dx}}{x} \qquad (6.2.9)$$

where SF is determined by Eq. (3.2.13).

The equations above are solved numerically and the grid system thus generated is shown in Fig. 6.3. In that figure it can be seen that the grid points are clustered in the middle of physical domain where the gradient of the solution is high.

6.2.3 Steep Gradients in Both Sides of the Spatial Domain

Suppose that the gradient of the solution for the problem being analyzed is known at some time level and is given by the following equations:

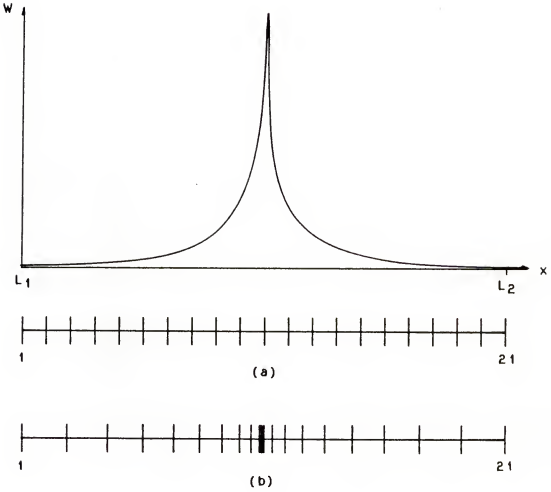


Figure 6.3 Grid system generated for problems with steep gradients in the middle of the spatial domain.
(a) Grid system before stretching.
(b) Grid system after stretching.

$$\frac{du}{dx} = 1.3(x - 1)^{-2.3} \qquad 1 \le x \le 11 \qquad (6.2.10)$$

$$\frac{du}{dx} = 1.3(21 - x)^{-2.3} \qquad 11 < x \le 21 \qquad (6.2.11)$$

Also suppose that the number of grid points to be used for solving this problem is fixed at IL=21 and that the boundaries of the spatial domain are at L_1=1 and L_2=21.

By substituting Eqs. (6.2.10) and (6.2.11) into Eq. (3.2.5) gives the weighting functions which are

$$W(x) = [\ 1 + |1.3(x - 1)^{-2.3}|\ ]^{SF} \qquad 1 \le x \le 11 \qquad (6.2.12)$$

$$W(x) = [\ 1 + |1.3(21 - x)^{-2.3}|\ ]^{SF} \qquad 11 < x \le 21 \qquad (6.2.13)$$

With the above weighting functions, the stretching function for this problem is given by Eq. (3.1.25); i.e., for $1 \le x \le 11$

$$K(x) = \frac{1 + 10\ \dfrac{\displaystyle\int_1^x [1+|1.3(x - 1)^{-2.3}|]^{-SF} dx}{\displaystyle\int_1^{11} 1+|1.3(x - 1)^{-2.3}|]^{-SF} dx}}{x} \qquad (6.2.14)$$

and for $11 \leq x \leq 21$

$$K(x) = \frac{11 + 10 \dfrac{\displaystyle\int_1^x [1+|1.3(x-1)^{-2.3}|]^{-SF} dx}{\displaystyle\int_{11}^{21} [1+|1.3(x-1)^{-2.3}|]^{-SF} dx}}{x} \qquad (6.2.15)$$

where SF is determined by Eq. (3.2.13).

The equations above are solved numerically and the grid system thus generated is shown in Fig. 6.4. In that figure it can be seen that the that the grid points are clustered toward the two boundaries of the spatial domain where the gradient of the solution is high.

These three examples shown the capability of the method developed to formulate stretching functions so that the grid points in the spatial domain are distributed where they are most needed.

6.3 Stretching Functions for 1-D Initial-Value Problems

In this section, the method for formulating stretching functions was used to generate a grid system for solving a one-dimensional initial-value problem. The problem was solved by using a finite-difference method and a grid system generated by a stretching function. The solution, thus obtained was compared with the solution obtained by using a grid system with equally spaced grid points, and the exact known solution.

The initial-value problem to be solved numerically is given by

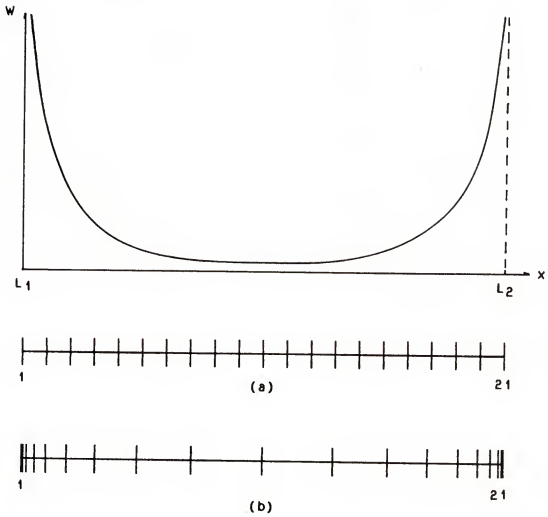


Figure 6.4 Grid system generated for problems with steep gradients in both sides of the spatial domain.
(a) Grid system before stretching.
(b) Grid system after stretching.

$$\frac{du}{dx} + 0.725\, x^{-1.725} = 0 \qquad (6.3.1)$$

subjected to the following condition:

$$u(x=1) = 1.0 \qquad (6.3.2)$$

The number of grid points used for solving this problem is IL=21 and the boundaries of the spatial domain are L_1=1 and L_2=11.

The solution of this problem was obtained by using the SAAGG method following the steps of the algorithm presented in section 3.3 of Chapter III. The first step in that algorithm is to transform Eq. (6.3.2) so that the ξ is the independent variable instead of x. The transformed equation for this problem in given by

$$\frac{du}{d\xi}\xi_x + 0.725\, [x(\xi)]^{-1.725} = 0 \qquad (6.3.3)$$

where $x(\xi)$ is determined by using Eq. (3.2.2) with the technique for generating stretching functions and the metric coefficient ξ_x is determined by using Eq. (3.3.8).

With the transformed equation given by Eq. (6.3.3), the next step is to use a finite-difference method to derive a finite difference equation (FDE). Here, a first-order-accurate, backward-difference approximation was used and the following FDE was obtained from Eq. (6.3.3):

$$\frac{u_i - u_{i-1}}{\Delta\xi}\xi_{x_i} + 0.725[x(\xi)]^{-1.725} = 0 \qquad (6.3.4)$$

The numerical solution of Eqs. (6.3.1) and (6.3.2) was obtained from Eq. (6.3.4) and the results are shown in Fig. 6.5. This figure, also

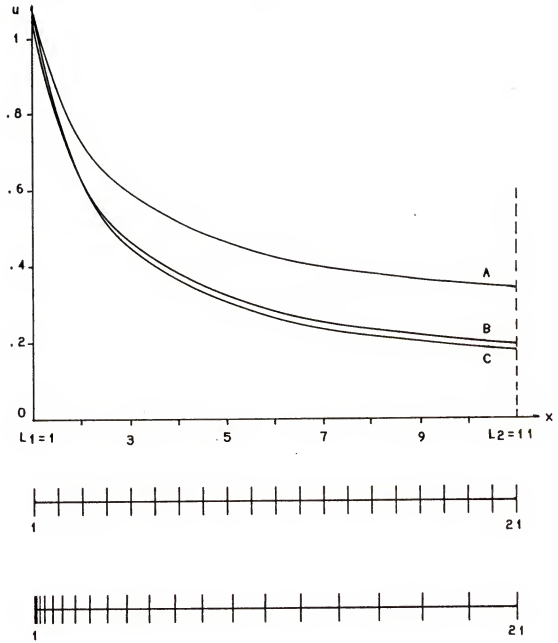


Figure 6.5 Solutions obtained for the initial-value problem given by Eqs.(6.3.1) and (6.3.2). The unit for u is m/sec and the unit for x is m.

(A) Numerical solution obtained by using a grid system with equally spaced grid points.

(B) Numerical solution obtained by using stretching function and the SAAGG method.

(C) Exact solution.

shows a comparison between the solution obtained by using the SAAGG method developed in this investigation, the exact solution, and the numerical solution obtained by using a grid system with equally spaced grid points. In this comparison, it can be seen that numerical errors were reduced when the problem was solved by using a stretching function for redistributing the grid points in the spatial domain. Here, it is noted that this improvement was obtained by using the same number of grid points and the same finite-difference method. In Appendix C, the computational efficiency of the SAAGG method is compared with the computational efficiency of methods using equally spaced grid points.

This concludes the applications of the technique for formulating stretching functions developed in Section 3.1 of Chapter III. The examples given in this section demonstrated the feasibility and usefulness of the technique investigated. In the next chapter, more examples are given for using this technique in conjunction with the SAAGG method for solving one-dimensional unsteady problems.

CHAPTER VII

RESULTS FOR ONE-DIMENSIONAL UNSTEADY PROBLEMS

In this chapter, the solution-adaptive, algebraic grid generation (SAAGG) method developed in Chapter III was used with finite-difference methods to obtain numerical solutions for two one-dimensional problems. The two problems analyzed were the one-dimensional linearized inviscid Burgers' equation and the quasi-linear inviscid Burgers' equation. The numerical solutions obtained for these two problems were compared with the exact solutions.

7.1 Linearized Inviscid Burgers' Equation

Burgers' equation has been used widely as a model equation for testing and comparing computational techniques in fluid dynamics. This is because it is mathematically simple; it describes some of the most important transport mechanisms in fluid dynamics; and for certain initial and boundary conditions, it is easy to obtain the exact solution in analytical form [30,31]. The one-dimensional Burgers' equation is

$$\frac{\partial u}{\partial t} + \frac{\partial F}{\partial x} = \nu \frac{\partial^2 u}{\partial x^2} \tag{7.1.1}$$

where F is a function of u, and ν is a constant.

The one-dimensional, linearized inviscid Burgers' equation is obtained from Eq. (7.1.1) by neglecting the second-order derivative term and setting the function F to cu, in which c is a constant. In this section, it is desired to use the SAAGG method along with a finite-difference method to obtain numerical solutions to the one-dimensional, linearized inviscid Burgers' equation given below:

$$\frac{\partial u}{\partial t} + c \frac{\partial u}{\partial x} = 0 \tag{7.1.2}$$

subjected to the following initial and boundary conditions:

$$u(x,0) = \begin{cases} \dfrac{2(4 - x)}{3} & 1 \le x \le 2.5 \\ 1 & x > 2.5 \end{cases} \tag{7.1.3}$$

$$u(L_1,t) = 2 \tag{7.1.4}$$

In the equations above, c=1m/sec and x varies from L_1=1 m to L_2=11 m.

In order to use the SAAGG method, it is necessary to follow the steps of the algorithm presented in Section 3.3 of Chapter III. The first step in the algorithm is to transform Eq. (7.1.2) so that ξ and τ are the independent variable instead of x and t. For the problem described by Eqs. (7.1.2) to (7.1.4), the transformation of the physical domain in x-t coordinate system to the computational domain in ξ-τ

coordinate system is given by

$$\xi = \xi(x,t) \tag{7.1.5}$$

$$\tau = t \tag{7.1.6}$$

and the partial derivatives with respect to x and t are related to the partial derivatives with respect to ξ and τ by

$$\frac{\partial}{\partial x} = \frac{\partial \xi}{\partial x}\frac{\partial}{\partial \xi} + \frac{\partial \tau}{\partial x}\frac{\partial}{\partial \tau} \tag{7.1.7}$$

$$\frac{\partial}{\partial t} = \frac{\partial \xi}{\partial t}\frac{\partial}{\partial \xi} + \frac{\partial \tau}{\partial t}\frac{\partial}{\partial \tau} \tag{7.1.8}$$

where

$$\frac{\partial \tau}{\partial x} = 0 \quad \text{and} \quad \frac{\partial \tau}{\partial t} = 1 \tag{7.1.9}$$

Substitution of Eqs. (7.1.7) to (7.1.9) into Eq. (7.1.2) gives the following transformed equation:

$$\left(\frac{\partial \xi}{\partial t}\frac{\partial}{\partial \xi} + \frac{\partial}{\partial \tau}\right) u + c\left(\frac{\partial \xi}{\partial x}\frac{\partial}{\partial \xi}\right) u = 0$$

which simplifies to

$$\frac{\partial u}{\partial \tau} + (\xi_t + c\xi_x)\frac{\partial u}{\partial \xi} = 0 \qquad (7.1.10)$$

With the governing equation transformed, the next step is to use a finite-difference method to derive a finite-difference equation (FDE). Here, the Lax-Wendroff method [29] was used and the following FDE was derived from Eq. (7.1.10):

$$u_i^{n+1} = u_i^n - \frac{\Delta\tau}{2\Delta\xi}(\xi_t + c\xi_x)_i (u_{i+1}^n - u_{i-1}^n) +$$

$$(\frac{\Delta\tau}{2\Delta\xi})^{(2)} (\xi_t + c\xi_x)_i^{(2)} (u_{i+1}^n - 2u_i^n + u_{i-1}^n) \qquad (7.1.11)$$

In the equation above, the superscript (2) denotes square and the superscript n and n+1 denote time levels. The metric coefficients ξ_x and ξ_t are obtained from Eqs. (3.3.8) and (3.3.9), respectively.

The next two steps of the algorithm are to specify the duration of interest (T) and the number of grid points (IL) to be employed. Here, T was set equal to 9 seconds and IL was set to 21. The remaining steps of the algorithm described in Section 3.3 are straightforward and are not repeated except for the procedure used to evaluate the weighting function.

The weighting function appearing in Eq. (3.2.5) need to be represented in terms of the ξ- and τ-coordinates. This can be done by using Eqs. (7.1.7) to (7.1.9) as shown below:

$$W(\xi,\tau) = (1 + |\frac{\partial u}{\partial \xi}\xi_x|)^{SF} \tag{7.1.12}$$

By replacing $\partial u/\partial \xi$ in the above equation by a second-order-accurate in space central-difference approximation, the finite difference form of the weighting function is obtained, and is

$$W(\xi,\tau)_i^n = [1 + |(\frac{u_{i+1}^n - u_{i-1}^n}{2\Delta\xi})\ \xi_{x_i}^n|]^{SF} \tag{7.1.13}$$

The weighting function in the algorithm described in Section 3.3 is evaluated by using the equation above.

The numerical solution obtained from Eq. (7.1.2) to (7.1.4) by using Eq. (7.1.11) and the grid system generated by the SAAGG method is shown in Fig. 7.1. In that figure, it can be seen that the grid points in physical domain are relocated after each time step to follow the steep gradient of the problem which moves as time progresses.

The SAAGG method is more efficient than methods based on grid systems using equally spaced grid points because less grid points are needed to achieve the same accuracy. In this problem, for instance, the number of grid points would be multiplied by five if a grid system using equally spaced grid points was used instead of the grid system generated by the SAAGG method.

Here, it is noted that the oscillations near the steep gradient are not due to the grid generation technique used, but are due to the dispersive nature of the Lax-Wendroff method chosen [23].

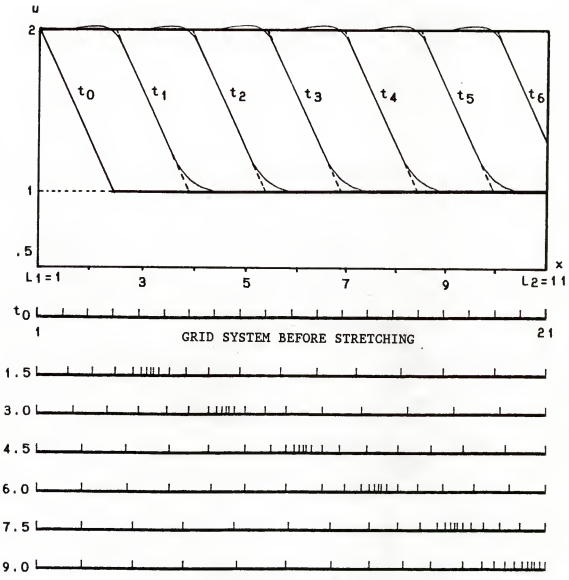


Figure 7.1 Solutions for the one-dimensional, linearized inviscid Burgers' equation given by Eqs. (7.1.1) to (7.1.3). The units for x, u and t are m, m/sec and sec, respectively.
—— Numerical solution obtained with the grid system generated by using the SAAGG method.
--- Exact solution.

The results given above demonstrate that the SAAGG method developed in this investigation is able to generate a solution-adaptive grid system for solving one-dimensional unsteady problems in which steep gradients can move about.

7.2 Quasi-linear Inviscid Burgers' Equation

Similar to the linearized, inviscid Burgers' equation, the quasi-linear inviscid Burgers' equation is also obtained from Eq. (7.1.1) by neglecting the second-order derivative term and setting the function F to $u^2/2$. In this section the SAAGG method is used along with a finite-difference method to obtain numerical solutions for the one-dimensional, quasi-linear inviscid Burgers' equation given below:

$$\frac{\partial u}{\partial t} + \frac{\partial}{\partial x}\left(\frac{u^2}{2}\right) = 0 \tag{7.2.1}$$

subjected to the following initial and boundary conditions:

$$u(x,0) = \begin{cases} 2 - \sin\left[\frac{\pi}{2}(x - 2)\right] & 1 \le x \le 3 \\ 1 & x > 3 \end{cases} \tag{7.2.2}$$

$$u_x(L_1,t) = 0 \tag{7.2.3}$$

where x varies between L_1=1 m and L_2=8.5 m.

In order to use the SAAGG method, it is necessary to follow the steps of the algorithm presented in Chapter III and already used in Section 7.1. The first step in the algorithm is to transform Eq.(7.2.1) so that ξ and τ are the independent variable instead of x and t. For the problem describe by Eqs. (7.2.1) to (7.2.3), the transformation of the physical domain in x-t coordinate system to the computational domain in ξ-τ coordinate system is given by Eqs. (7.1.5) to (7.1.6), and the partial derivatives with respect to x and y are related to the partial derivatives with respect to ξ and τ by Eqs. (7.1.7) to (7.1.9). Thus, substitution of Eqs.(7.1.5) to (7.1.9) into Eq. (7.2.1) gives the following transformed equation.

$$\frac{\partial u}{\partial \tau} + \xi_t \frac{\partial u}{\partial \xi} + \xi_x \frac{\partial}{\partial \xi}\left(\frac{u^2}{2}\right) = 0 \tag{7.2.4}$$

With the governing equation transformed, the next step is to use a finite-difference method to derive a finite-difference equation (FDE). Here, the Lax-Wendroff method [29] was used and the following FDE was derived from Eq. (7.2.4):

$$u_i^{n+1} = u_i^n - \xi_t \frac{\Delta\tau}{2\Delta\xi}(u_{i+1} - u_{i-1})^n - \xi_x \frac{\Delta\tau}{4\Delta x}(u_{i+1}^{(2)} - u_{i-1}^{(2)})^n + \tag{7.2.5}$$

$$\frac{\xi_x^{(2)}}{8}\left(\frac{\Delta\tau}{\Delta\xi}\right)^{(2)}[(u_i + u_{i+1})^n (u_{i+1}^{(2)} - u_i^{(2)})^n - (u_i + u_{i-1})^n (u_i^{(2)} - u_{i-1}^{(2)})^n]$$

In the equation above, the superscript (2) denotes square and the superscipt n and n+1 denote time levels. The metric coefficients ξ_x and ξ_t are obtained from Eqs. (3.3.8) and (3.3.9), respectively.

The next two steps of the algorithm are to specify the duration of interest (T) and the number of grid points (IL) to be employed. Here, T was set equal to 6 seconds and IL was set to 31. The same procedure used in Section 7.1 to evaluate the weighting function in terms of the ξ- and r-coordinates is used in this problem. The remaining steps of the algorithm described in Chapter III are straightforward and are not repeated here.

The numerical solution obtained from Eq. (7.2.1) to (7.2.3) by using Eq. (7.2.5) and the grid system generated by the SAAGG method are shown in Fig. 7.2. In this figure, it can be seen that the grid points in physical domain are relocated after each time step to follow the steep gradient of the problem which moves as time progresses. This problem differs from the problem solved in Section 7.1 because, in this case, the maximum gradient in the solution becomes steeper at each time step. Here, it is noted that the same remarks on the advantages of the SAAGG method, made when solving the linearized inviscid Burgers' equation, are valid for this problem.

To further demonstrate the usefulness of the SAAGG method, it was used along with the Lax-Wendroff method to obtain solutions for the one-dimensional, quasi-linear inviscid Burgers' equation defined by Eq. (7.2.1), subjected to another initial condition. This initial condition is given by the following equation:

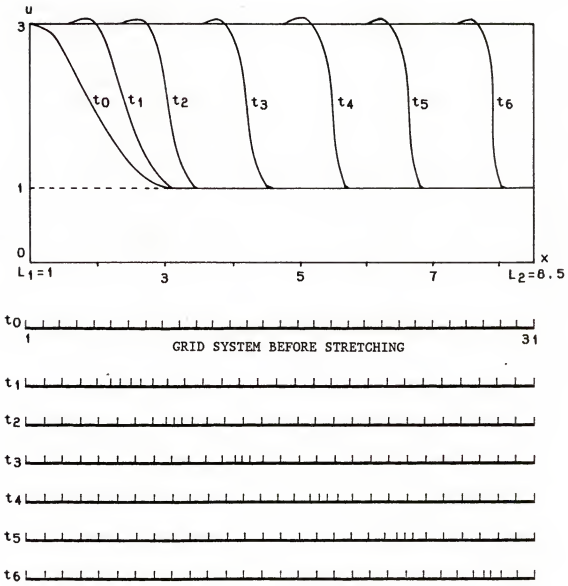


Figure 7.2 Solution of the one-dimensional, quasi-linear inviscid Burgers' equation given by Eqs. (7.2.1) and (7.2.3) obtained with the grid system generated by using the SAAGG method. The units for x, u and t are m, m/sec and sec, respectively.

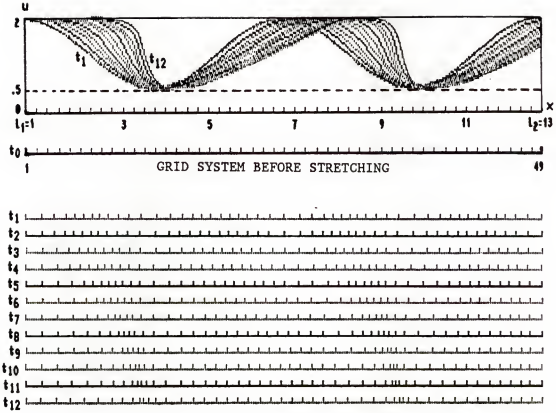


Figure 7.3 Solution of the one-dimensional, quasi-linear inviscid Burgers' equation given by Eqs. (7.2.1) and (7.2.6) obtained with the grid system generated by using the SAAGG method. The units for x, u and t are m, m/sec and sec, respectively.

$$u(x,0) = 1.25 + 0.75\sin[\frac{\pi}{3}(x + 0.5)] \quad x \geq 1 \qquad (7.2.6)$$

where x varies from L_1=1 m and L_2=13 m, the number of grid points is IL=49 and the other conditions and specifications remain the same.

By using the initial condition given above, the capability of the SAAGG method for generating grid systems to follow steep gradients in more than one region in the physical domain is demonstrated.

The numerical solution obtained from Eq. (7.2.1) subjected to the initial condition above and the grid generated by the SAAGG method are shown in Fig. 7.3. In this figure, it can be seen that the grid points in physical domain are relocated after each time step to follow the steep gradients in the physical domain. These results demonstrate the capability of the SAAGG method to generate grid systems to follow more than one steep gradient.

The examples above conclude the demonstrations of the SAAGG method developed in this investigation for solving one-dimensional unsteady problems.

CHAPTER VIII

RESULTS FOR TWO-DIMENSIONAL UNSTEADY PROBLEMS

In this chapter, the solution-adaptive, algebraic grid generation (SAAGG) method developed in Chapter IV and V was used with finite-difference methods to obtain numerical solutions for two-dimensional problems. This chapter is divided in two sections. In the first section, the SAAGG method was used to solve two two-dimensional problems with rectangularly shaped spatial domains. In the second section, the SAAGG method was used in conjunction with the Two-Boundary Technique to generate a grid system in a convergent-divergent spatial domain.

8.1 Problems With Rectangularly Shaped, Spatial Domain

In this section, two problems with rectangularly shaped spatial domain were solved by using the SAAGG method with finite-difference methods. The two problems analyzed were the two-dimensional, unsteady problems of plane wave propagation and the propagation of concentric, circular waves in the radial direction.

8.1.1 Plane Wave Propagation

The two-dimensional, quasi-linear inviscid Burgers' equation was used with appropriate initial and boundary conditions to model a plane wave propagating in a direction 45 degrees from the x-axis. These

equations are given below:

$$\frac{\partial u}{\partial t} + \frac{\partial}{\partial x}\left(\frac{u^2}{2}\right) + \frac{\partial}{\partial y}\left(\frac{u^2}{2}\right) = 0 \tag{8.1.1}$$

$$u(x,y,0) = 2 + \sin\frac{\pi(x+y)}{4} \tag{8.1.2}$$

$$u_x(L_1, L_1', t) = 0 \tag{8.1.3}$$

$$u_x(L_1, L_1', t) = 0 \tag{8.1.4}$$

where x varies between L_1=1 m and L_2=7 m, and y varies between L_1'=1 m and L_2'=7 m.

To use the SAAGG method, it is necessary to follow the steps of the algorithm presented in Chapter IV. The first step is to transform Eq. (8.1.1) so that ξ, η and τ are the independent variable instead of x, y and t. For the problem described by Eqs. (8.1.1) to (8.1.4), the transformation of the physical domain in x-y-t coordinate system to the computational domain in ξ-η-τ coordinate system is given by Eq. (2.3) and the partial derivatives with respect to x, y and t are related to the partial derivatives with respect to ξ, η and τ by Eq. (2.4). Thus, substitution of Eqs. (2.3) and (2.4) into Eq. (8.1.1) gives the following transformed equation:

$$\xi_t \frac{\partial u}{\partial \xi} + \eta_t \frac{\partial u}{\partial \eta} + \frac{\partial u}{\partial \tau} + (\xi_x + \xi_y) \frac{\partial}{\partial \xi} \left(\frac{u^2}{2}\right) +$$

$$(\eta_x + \eta_y) \frac{\partial}{\partial \eta} \left(\frac{u^2}{2}\right) = 0 \qquad (8.1.5)$$

The equation above can be written more compactly as follows:

$$\frac{\partial u}{\partial \tau} = - \frac{\partial F}{\partial \xi} - \frac{\partial G}{\partial \eta} \qquad (8.1.6)$$

where

$$F = \xi_t u + (\xi_x + \xi_y) \frac{u^2}{2} \qquad (8.1.7)$$

$$G = \eta_t u + (\eta_x + \eta_y) \frac{u^2}{2} \qquad (8.1.8)$$

With the governing equation transformed, the next step in the algorithm is to use a finite difference method to derive a finite-difference equation (FDE). Here, the Lax-Wendroff method [29] was used and the following FDE was derived from Eq. (8.1.6):

$$u^{n+1}_{i,j} = u^{n}_{i,j} - \frac{\Delta\tau}{\Delta\xi}\frac{F^{n}_{i+1,j} - F^{n}_{i-1,j}}{2} - \frac{\Delta\tau}{\Delta\eta}\frac{G^{n}_{i,j+1} - G^{n}_{i,j-1}}{2} +$$

$$\frac{1}{2}\left(\frac{\Delta\tau}{\Delta\xi}\right)^{(2)}\left[A_{i+1/2}(F^{n}_{i+1,j} - F^{n}_{i,j}) - A_{i-1/2}(F^{n}_{i,j} - F^{n}_{i-1,j})\right] +$$

$$\frac{1}{2}\left(\frac{\Delta\tau}{\Delta\eta}\right)^{(2)}\left[B_{i+1/2}(G^{n}_{i,j+1} - G^{n}_{i,j}) - B_{j-1/2}(G^{n}_{i,j} - G^{n}_{i,j-1})\right] \qquad (8.1.9)$$

where

$$F^{n}_{i,j} = \xi_t u^{n}_{i,j} + (\xi_x + \xi_y)\frac{u^{n(2)}_{i,j}}{2} \qquad (8.1.10)$$

$$G^{n}_{i,j} = \eta_t u^{n}_{i,j} + (\eta_x + \eta_y)\frac{u^{n(2)}_{i,j}}{2} \qquad (8.1.11)$$

$$A_{i\pm1/2} = \xi_t + \frac{1}{2}(\xi_x + \xi_y)(u^{n}_{i,j} + u^{n}_{i\pm1,j}) \qquad (8.1.12)$$

$$B_{j\pm1/2} = \eta_t + \frac{1}{2}(\eta_x + \eta_y)(u^{n}_{i,j} + u^{n}_{i,j\pm1}) \qquad (8.1.13)$$

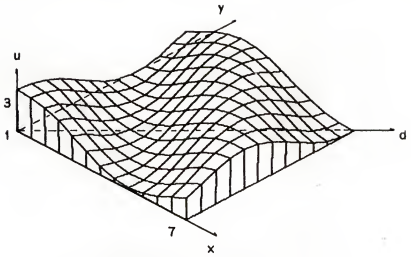
$$F^{n}_{i\pm1,j} = \xi_t \, u^{n}_{i\pm1,j} + (\xi_x + \xi_y)\frac{u^{(2)}_{i\pm1,j}}{2} \tag{8.1.14}$$

$$G^{n}_{i,j\pm1} = \eta_t \, u^{n}_{i,j\pm1} + (\eta_x + \eta_y)\frac{u^{(2)}_{i,j\pm1}}{2} \tag{8.1.15}$$

In the equation above, the superscript (2) denotes square and the superscripts n and n+1 denote time levels. The metric coefficients ξ_x, ξ_y, η_x, η_y, ξ_t, η_t, and τ_t are obtained from Eqs. (2.6) to (2.12), respectively.

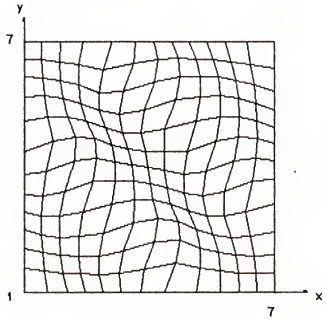
The next two steps of the algorithm are to specify the duration of interest (T) and the number of grid points (IL) and (JL) to be employed. Here, T was set equal to 6 seconds and IL and JL were both set to 13. The remaining steps of the algorithm described in Section 4.1 are straightforward and are not repeated here. The weighting function in the ξ-direction is obtained from Eq. (7.1.13). The weighting function in the η-direction is also obtained from Eq. (7.1.13) by replacing the coordinate ξ by η.

The numerical solution obtained from Eqs. (8.1.1) to (8.1.4) by using Eq. (8.1.9) and the grid system generated by the SAAGG method are shown in Figs. 8.1 to 8.5, where the unit for x and y is m and the unit for u depends on the variable being analyzed. In these figures, it can be seen that the grid points in the two-dimensional physical domain are relocated after each time step to follow the steep gradient of the plane wave as it travels.



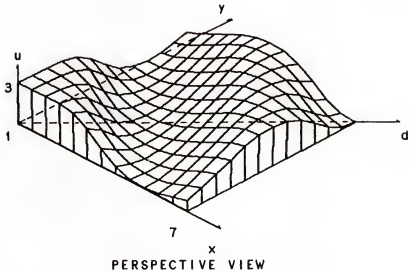
PERSPECTIVE VIEW

SIDE VIEW



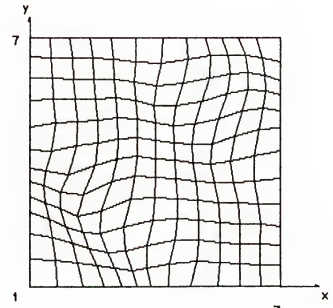
SAAGG 2-D GRID

Figure 8.1 Solution for the propagation of a plane wave obtained by using the grid system generated by the SAAGG method at t=1 sec.



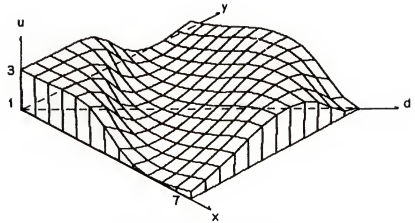
PERSPECTIVE VIEW

SIDE VIEW



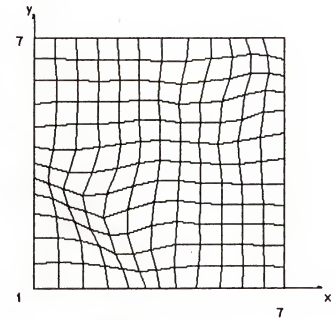
SAAGG 2-D GRID

Figure 8.2 Solution for the propagation of a plane wave obtained by using the grid system generated by the SAAGG method at t=1.5 sec.



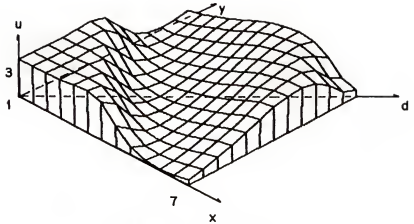
PERSPECTIVE VIEW

SIDE VIEW



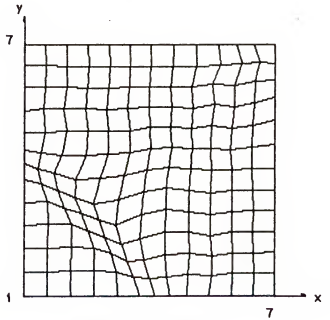
SAAGG 2-D GRID

Figure 8.3 Solution for the propagation of a plane wave obtained by using the grid system generated by the SAAGG method at t=2 sec.



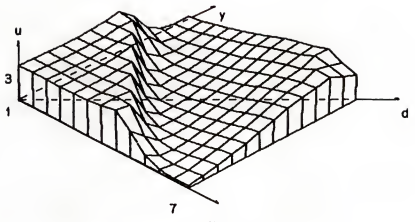
PERSPECTIVE VIEW

SIDE VIEW



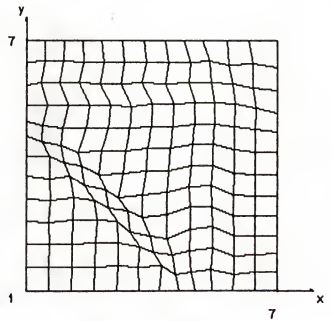
SAAGG 2-D GRID

Figure 8.4 Solution for the propagation of a plane wave obtained by using the grid system generated by the SAAGG method at t=3 sec.



PERSPECTIVE VIEW

SIDE VIEW



SAAGG 2-D GRID

Figure 8.5 Solution for the propagation of a plane wave obtained by using the grid system generated by the SAAGG method at t=6 sec.

The remarks made in Section 7.1 for the one-dimensional problems are also valid for the two-dimensional problems discussed in this section. Again, using the grid system generated by the SAAGG method is more efficient than using a grid system with equally spaced grid points to achieve the same desired accuracy in the solution. In this problem, for instance, the number of grid points would be multiplied by sixteen if a grid system with equally spaced grid points was used instead of the grid system generated by the SAAGG method. In Appendix C, the SAAGG method is compared with an equally spaced method in terms of efficiency.

As was noted for one-dimensional problems, the oscillations along the steep gradient are not due to the grid generation technique used, but to the dispersive nature of the Lax-Wendroff method chosen [23]. Here, it is noted that the steep gradient tends to smear near the boundaries after several time steps. This is due to the first-order-accurate finite-difference approximation used to compute the solution at the boundaries.

The results obtained in this section demonstrate that the SAAGG method can generate solution-adaptive grid systems for analyzing two-dimensional unsteady problems with steep gradients which move about as time progresses.

8.1.2 Circular Wave Propagation

The two-dimensional, quasi-linear inviscid Burgers' equation was used with appropriated initial and boundary conditions to model the propagation of concentric circular waves in the radial direction. These equations are given below:

$$\frac{\partial u}{\partial t} + \frac{\partial}{\partial x}\left(\frac{u^2}{2}\right) + \frac{\partial}{\partial y}\left(\frac{Eu^2}{2}\right) = 0 \qquad (8.1.16)$$

$$u(x,y,0) = 2 + \sin \frac{\pi[2(x^2 + y^2)]^{1/2}}{4} \qquad (8.1.17)$$

$$u_x(L_1, L_1', t) = 0 \qquad (8.1.18)$$

$$u_x(L_1, L_1', t) = 0 \qquad (8.1.19)$$

where

$$E = \frac{y - L_1'}{x - L_1} \qquad (8.1.20)$$

In the above equations, x varies between L_1=1 m and L_2=7 m, and y varies between L_1'=1 m and L_2'=7 m.

To use the SAAGG method, it is necessary to follow the steps of the algorithm presented in Chapter IV. The first step in the algorithm is to transform Eq. (8.1.16) so that ξ, η and τ are the independent variable instead of x, y and t. For the problem describe by Eqs. (8.1.16) to (8.1.20), the transformation of the physical domain in x-y-t coordinate system to the computational domain in ξ-η-τ coordinate system is given by Eq. (2.3) and the partial derivatives with respect to x, y and t are related to the partial derivatives with respect to ξ, η and τ

by Eq. (2.4). Thus, the substitution of Eqs. (2.3) and (2.4) into Eq. (8.1.16) gives the following transformed equation:

$$\xi_t \frac{\partial u}{\partial \xi} + \eta_t \frac{\partial u}{\partial \eta} + \frac{\partial u}{\partial \tau} + (\xi_x + E\xi_y) \frac{\partial}{\partial \xi}\left(\frac{u^2}{2}\right) +$$

$$(\eta_x + E\eta_y) \frac{\partial}{\partial \eta}\left(\frac{u^2}{2}\right) + \frac{u^2}{2(x - L_1)} = 0 \qquad (8.1.21)$$

The equation above can be written more compactly as follows:

$$\frac{\partial u}{\partial \tau} = - \frac{\partial F}{\partial \xi} - \frac{\partial G}{\partial \eta} - S \qquad (8.1.22)$$

where

$$F = \xi_t u + (\xi_x + E\xi_y) \frac{u^2}{2} \qquad (8.1.23)$$

$$G = \eta_t u + (\eta_x + E\eta_y) \frac{u^2}{2} \qquad (8.1.24)$$

$$S = \frac{u^2}{2(x - L_1)} \qquad (8.1.25)$$

With the governing equation transformed, the next step in the

algorithm is to use a finite difference method to derive a finite-difference equation (FDE). Here, the Lax-Wendroff method [29] was used and the following FDE was derived from Eq. (8.1.22):

$$u^{n+1}_{i,j} = u^{n}_{i,j} - \frac{\Delta\tau}{\Delta\xi}\frac{F^{n}_{i+1,j} - F^{n}_{i-1,j}}{2} - \frac{\Delta\tau}{\Delta\eta}\frac{G^{n}_{i,j+1} - G^{n}_{i,j-1}}{2} +$$

$$\frac{1}{2}\left(\frac{\Delta\tau}{\Delta\xi}\right)^{(2)}\left[A_{i+1/2}(F^{n}_{i+1,j} - F^{n}_{i,j}) - A_{i-1/2}(F^{n}_{i,j} - F^{n}_{i-1,j})\right] +$$

$$\frac{1}{2}\left(\frac{\Delta\tau}{\Delta\eta}\right)^{(2)}\left[B_{i+1/2}(G^{n}_{i,j+1} - G^{n}_{i,j}) - B_{j-1/2}(G^{n}_{i,j} - G^{n}_{i,j-1})\right] +$$

$$\frac{\Delta\tau u^{2}}{2(x - L_1)} \qquad (8.2.26)$$

where

$$F^{n}_{i,j} = \xi_t u^{n}_{i,j} + (\xi_x + E_i\xi_y)\frac{u^{n(2)}_{i,j}}{2} \qquad (8.1.27)$$

$$G^{n}_{i,j} = \eta_t u^{n}_{i,j} + (\eta_x + E_i\eta_y)\frac{u^{n(2)}_{i,j}}{2} \qquad (8.1.28)$$

$$A_{i\pm1/2} = \xi_t + \frac{1}{2}(\xi_x + E_i\xi_y)(u^n_{i,j} + u^n_{i\pm1,j}) \tag{8.1.29}$$

$$B_{j\pm1/2} = \eta_t + \frac{1}{2}(\eta_x + E_i\eta_y)(u^n_{i,j} + u^n_{i,j\pm1}) \tag{8.1.30}$$

$$F^n_{i\pm1,j} = \xi_t\, u^n_{i\pm1,j} + (\xi_x + E_i\xi_y)\frac{u^{n(2)}_{i\pm1,j}}{2} \tag{8.1.31}$$

$$G^n_{i,j\pm1} = \eta_t\, u^n_{i,j\pm1} + (\eta_x + E_i\eta_y)\frac{u^{n(2)}_{i,j\pm1}}{2} \tag{8.1.32}$$

In the equation above, the superscript (2) denotes square and the superscripts n and n+1 denote time levels. The metric coefficients ξ_x, ξ_y, η_x, η_y, ξ_t, η_t, and τ_t are obtained from Eqs. (2.6) to (2.12).

The next two steps of the algorithm are to specify the duration of interest (T) and the number of grid points (IL) and (JL) to be employed. Here, T was set equal to 6 seconds and IL and JL were set equal to 13. The remaining steps of the algorithm described in Section 4.1 are straightforward and are not repeated here. The weighting function in the ξ-direction is obtained from Eq. (7.1.13). The weighting function in the η-direction is also obtained from Eq. (7.1.13) by replacing the coordinate ξ by η.

The numerical solution obtained from Eq. (8.1.16) to (8.1.20) by using Eq. (8.1.26) and the grid system generated by the SAAGG method are shown in Figs. 8.6 to 8.8. In these figures it can be seen that the

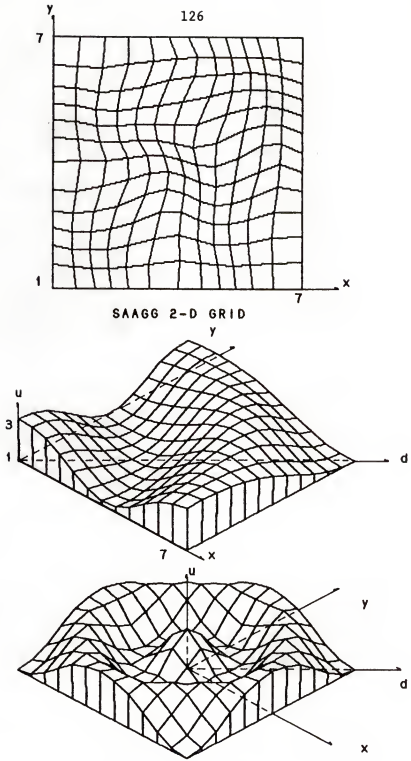


Figure 8.6 Solution for the propagation of concentric circular waves obtained by using the grid system generated by the SAAGG method at t=2 sec.

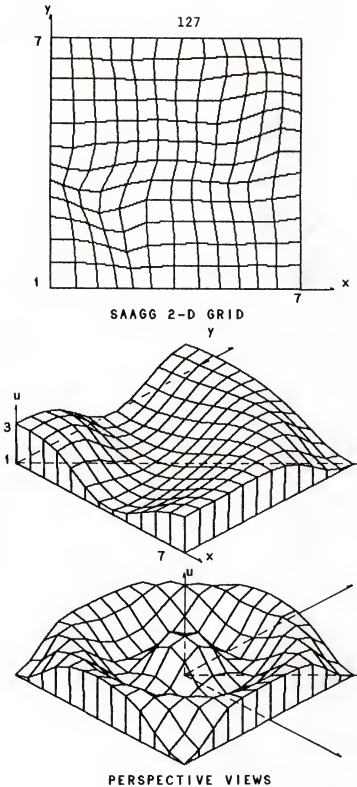


Figure 8.7 Solution for the propagation of concentric circular waves obtained by using the grid system generated by the SAAGG method at t=4 sec.

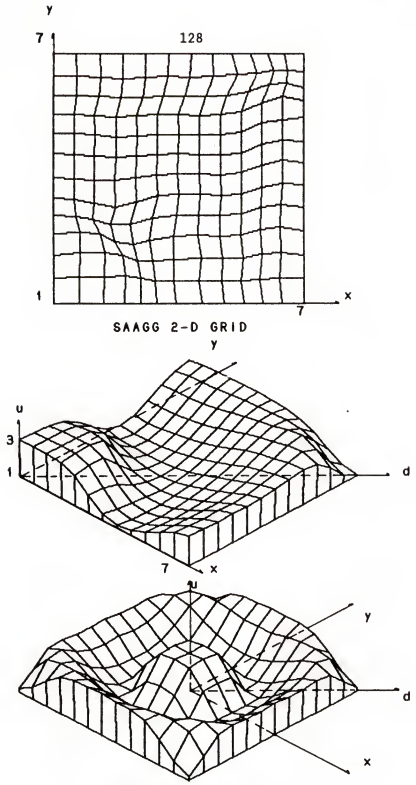


Figure 8.8 Solution for the propagation of concentric circular waves obtained by using the grid system generated by the SAAGG method at t=6 sec.

grid points in the two-dimensional physical domain are relocated after each time step by the SAAGG method to follow the steep gradient of the circular wave travelling in the radial direction. This example concludes the demonstration of the SAAGG method developed in Chapter IV for problems with rectangularly shaped spatial domain.

8.2 Problems With Arbitrarily Shaped, Spatial Domains

In this section, the solution-adaptive, algebraic grid generation (SAAGG) method developed in Chapter V is used with the Two-Boundary Technique to generate a grid system in a convergent-divergent spatial domain.

In order to use the SAAGG method, it is necessary to follow the steps of the algorithm presented in Section 5.3 of Chapter V. Here, the steps 1, 2 and 4 of the algorithm are not applied because it is desired to generate a grid system in a spatial domain for a problem in which the solution is known at a given time level. In step 3, the number of grid lines were specified as IL=21 and JL=13, and in step 5, the correspondence between the boundaries of the spatial domain and the computational domain is shown in Fig. 8.9.

In the next step of the algorithm, the coordinates of the two boundaries AD and BC shown in Fig. 8.9 and expressed by Eqs. (5.1.1.) to (5.1.4) are given by

$$x_1(\xi,0) = (L_2 - L_1)\xi + L_1 \tag{8.2.1}$$

$$y_1(\xi,0) = V - (V - 1)(2\xi - 1)^2 \tag{8.2.2}$$

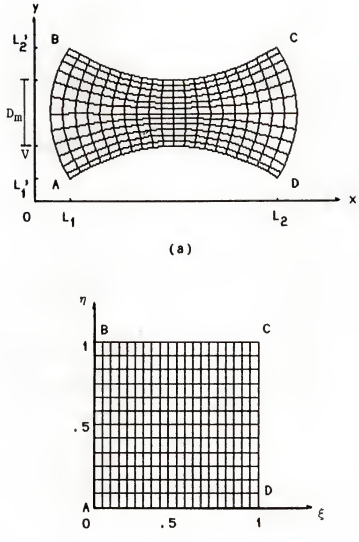


Figure 8.9 Correspondence between the boundaries of the physical domain and the computational domain.
(a) convergent-divergent shaped physical domain (x-y-t).
(b) computational domain (ξ-η-τ).

$$x_2(\xi,1) = (L_2 - L_1)\xi + L_1 \tag{8.2.3}$$

$$y_2(\xi,1) = (V + D_m) - (V - 1)(2\xi - 1)^2 \tag{8.2.4}$$

where $L_1 = 1$ cm and $L_2 = 9$ cm, $V=2.25$ cm and $D_m=2.5$ cm.

The partial derivatives appearing in Eqs. (5.2.1) and (5.2.2) are obtained from Eqs. (8.2.1) to (8.2.4) and are

$$A_2 = A_1 = L_2 - L_1 = 8 \tag{8.2.5}$$

$$B_2 = -B_1 = 4(V - 1)(2\xi - 1) = 5(2\xi - 1) \tag{8.2.6}$$

With the partial derivatives obtained and the coordinates Q and P obtained from Eqs. (5.2.7) to (5.2.8), the coefficients K_i are determined from Eqs. (5.2.11) and (5.2.12) for each grid line ξ_i and are given by

$$K_i = \frac{Q - x_1}{(V-1)(2\xi_i-1)} \qquad i=2,3,\ldots,IL-1 \tag{8.2.7}$$

These coefficients are compared with the criteria for nonoverlapping evaluated by using Eq. (5.2.31) with $D=D_m=2.5$ cm and $A=L_2-L_1=8$ cm; i.e.,

$$0 \leq K_i \leq 0.937 \tag{8.2.8}$$

At this point, the initial grid system can be determined by using

Eqs. (5.2.1) and (5.2.2) with the coefficients K_i given by Eq. (8.2.5), the partial derivatives A_1, A_2, B_1 and B_2 given by Eqs. (8.2.5) and (8.2.6), and the coefficients h_1, h_2, h_3 and h_4 given by Eqs. (5.1.7) to (5.1.10). With this initial grid system the coefficients Ka, Kb and Kc can be determined by using Eq. (5.2.21) and are

$$Ka \approx 0.316 \tag{8.2.9}$$

$$Kb \approx 0.567 \tag{8.2.10}$$

$$Kc \approx 0.937 \tag{8.2.11}$$

The coefficients determined above are compared with the coefficients K_i obtained from Eq. (8.2.7) to ensure that every grid line is clustering toward the boundaries by using the following inequality:

$$0.567 \leq K_i \leq 0.937 \tag{8.2.12}$$

In the next step of the algorithm, the deviation from orthogonality is specified as β=0.9 degrees. This value is compared to the values of θ_i obtained by using Eq. (5.2.40). In this problem, every θ was found to be less than β. Thus, the algorithm proceeds from step 19.

In step 19 of the algorithm, the gradient of velocity U_y and the gradient of pressure P_x were used with Eq. (3.2.5) to determine the weighting functions W(y) and W(x), respectively, to be used with the SAAGG method for determining the coordinates of the grid points in the spatial domain.

The grid system generated by using the SAAGG method is shown in Fig. 8.10 for two different profiles of pressure inside the spatial domain.

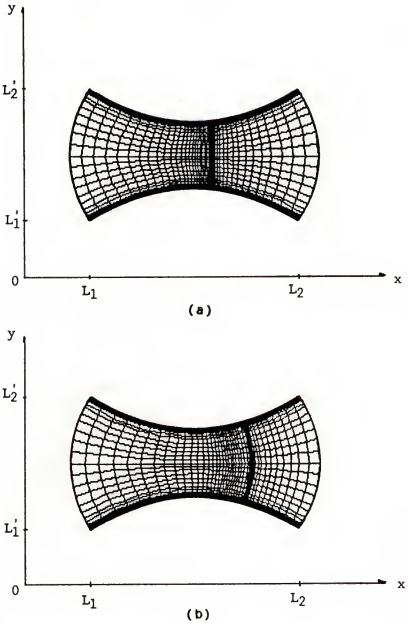


Figure 8.10 Grid system generated by using the SAAGG method inside a convergent-divergent-shaped spatial domain for two different profiles of pressure.
(a) Mach number 1.5.
(b) Mach number 2.0.

In this figure, it can be seen that the grid points in the spatial domain are clustered to regions of steep gradients; i.e., near the region of discontinuity of pressure in the x-direction and near the region of high gradients of velocity at boundaries in the y-direction. Here, it is noted that the SAAGG method was able to determine the coordinates of grid points independently in each coordinate direction by using different weighting functions, namely, the pressure gradient in the x-direction and the velocity gradient in the y-direction.

The examples above conclude the demonstration of the SAAGG method developed in this investigation for generating solution-adaptive grid systems in arbitrarily shaped spatial domains.

CHAPTER IX

CONCLUSIONS AND RECOMMENDATIONS

In this investigation, a solution-adaptive, algebraic grid generation (SAAGG) method was developed which can generate non-uniform grid systems needed by finite-difference algorithms to obtain numerical solutions for fluid flow problems. The grid systems generated by the SAAGG method are designed especially for unsteady problems with disparate length scales in different parts of the spatial domain which can move about in an unpredictable manner.

The SAAGG method is based on a new class of stretching functions developed in this investigation which are derived from a variational principle. These stretching functions couple the distribution of the grid points in the physical domain to the numerical solution being computed. That technique for formulating stretching functions also can be used for formulating stretching functions which are not coupled to the solution for use with any non-adaptive grid generation method. When using stretching functions which are coupled to the solution for designing the SAAGG method, parameters controlling the smoothness and orthogonality of the grid system were added to the grid generation process. This makes the SAAGG method more useful in generating grid systems to be used with finite-difference methods.

The SAAGG method was applied to several problems and these applications demonstrated that the SAAGG method can redistribute the grid points in the physical domain where they are most needed to reduce numerical errors. By placing the grid points where most needed the total number of grid points necessary to obtain solutions of a desired accuracy can be reduced considerably and this enhance computational efficiency as shown in Appendix C.

As with any other method, improvements and extensions can be made. For the SAAGG method developed is this investigation, the following improvement, extension and aplication are suggested in future works:

1. An improvement in the technique developed for formulating stretching functions so that smoothness, orthogonality and clustering can all be accounted for by a variational principle.
2. An extension of the SAAGG method so that it can generate solution-adaptive grid systems in 3-D spatial domains.
3. An application of the SAAGG method so that it can be combined with the Four-Boundary Technique to generate solution-adaptive grid systems in which four boundaries of the spatial domain can be account in the formulation.

APPENDIX A

DERIVATION OF THE EULER-LAGRANGE EQUATION

The Euler-Lagrange equation is a partial derivative equation which once solved leads to a function y(x) which is a solution for an integral problem of minimization. The problem is to determine a function y(x) so that an integral between two limits x_0 and x_1 is a minimum. This integral is

$$I = \int_{x_0}^{x_1} F(x,y,y')\, dx \tag{A.1}$$

where F is a functional formed by the independent variable x, the dependent variable y and its first derivative y'.

By assuming that y(x) is the solution for the problem, the following substitution can be made in Eq. (A.1):

$$\bar{y}(x) = y(x) + \alpha\, g(x) \tag{A.2}$$

where α is a constant between $(-\infty,+\infty)$ and g(x) is a function so that $g(x_0)=g(x_1)=0$. Then, Eq. (A.1) can be written as follows:

$$I(\alpha) = \int_{x_0}^{x_1} F(x,y + \alpha g, y' + \alpha g')\, dx \tag{A.3}$$

When $\alpha = 0$ the integral above is a minimum because in this case $\bar{y}(x) = y(x)$ is the solution for the problem. Then the variation of Eq. (A.3) is zero; i.e.,

$$\delta I(\alpha) = \frac{\partial I}{\partial \alpha} \delta\alpha = 0 \tag{A.4}$$

The substitution of Eq. (A.3) into Eq. (A.4) yields

$$\delta I(\alpha) = \int_{x_0}^{x_1} \frac{\partial}{\partial \alpha}[F(x,\bar{y},\bar{y}')] \, \delta\alpha \, dx \tag{A.5}$$

In the above equation the partial derivative can be evaluate as follows:

$$\frac{\partial F}{\partial \alpha} = \frac{\partial F}{\partial \bar{y}} \frac{\partial \bar{y}}{\partial \alpha} + \frac{\partial F}{\partial \bar{y}'} \frac{\partial \bar{y}'}{\partial \alpha} = \frac{\partial F}{\partial \bar{y}} g(x) + \frac{\partial F}{\partial \bar{y}'} g'(x) \tag{A.6}$$

Substitution of Eq. (A.6) into Eq. (A.5) yields

$$\delta I(\alpha) = \int_{x_0}^{x_1} .[\frac{\partial F}{\partial x} g(x) + \frac{\partial F}{\partial y'} g'(x)] \, \delta\alpha \, dx = 0 \tag{A.7}$$

In Eq. (A.7) the second term can be integrated by parts, as follows:

$$\int_{x_0}^{x_1} \frac{\partial F}{\partial y'} g'(x)dx = [\frac{\partial F}{\partial y'} g(x)]_{x_0}^{x_1} - \int_{x_0}^{x_1} g(x)\frac{d}{dx}(\frac{\partial F}{\partial y'}) \, dx \tag{A.8}$$

The first term on the right hand side of Eq. (A.8) is zero because $g(x_0)=g(x_1)=0$ by hypothesis. Then, the substitution of Eq. (A.8) into Eq. (A.7) yields

$$\delta I(\alpha) = \int_{x_0}^{x_1} g(x)\left[\frac{\partial F}{\partial y} - \frac{d}{dx}\left(\frac{\partial F}{\partial y'}\right)\right] \delta\alpha \, dx = 0 \qquad (A.9)$$

The above equation is satisfied when the term inside the brackets is zero; i.e.,

$$\frac{\partial F}{\partial y} - \frac{d}{dx}\left(\frac{\partial F}{\partial y'}\right) = 0 \qquad (A.10)$$

The above equation is known as the Euler-Lagrange equation related to the integral minimization problem of Eq. (A.1).

APPENDIX B

EXACT SOLUTION OF THE LINEARIZED INVISCID BURGERS' EQUATION

The linearized inviscid Burgers' equations was presented in Section 7.1 and it is

$$\frac{\partial u}{\partial t} + c \frac{\partial u}{\partial x} = 0 \tag{B.1}$$

where c is a positive constant.

The exact solution to Eq. (B.1) can be obtained by assuming continuous solutions u=u(t,x), so that the total derivative of u to the variable t exists; i.e.,

$$\frac{du}{dt} = \frac{\partial u}{\partial t} + \frac{\partial u}{\partial x} \frac{dx}{dt} \tag{B.2}$$

By letting

$$\frac{dx}{dt} = c \tag{B.3}$$

the Eq. (B.2) can be written as follows:

$$\frac{du}{dt} = \frac{\partial u}{\partial t} + c \frac{\partial u}{\partial x} \tag{B.4}$$

The substitution of Eq. (B.4) into Eq. (B.1) gives the following ordinary differential equation (ODE).

$$\frac{du}{dt} = 0 \tag{B.5}$$

The above equation implies that along any characteristic curve defined by Eq. (B.3), Eq. (B.1) becomes an ODE.

The integration of Eq. (B.3) gives

$$\int_{\xi}^{x} dx = c \int_{0}^{t} dt$$

$$x = \xi + ct \tag{B.6}$$

where ξ is can be any chosen x-coordinate at time t=0. The integration of Eq. (B.5) gives

$$\int_{u(\xi)}^{u} du = 0$$

$$u = u(\xi) \tag{B.7}$$

where $u(\xi)$ is the solution at the coordinate $x=\xi$ and time t=0.

Equation (B.7) shows that along each characteristic curve given by Eq. (B.6) the solution u does not change and can be determined once the initial and boundary conditions of the problem are known.

For the problem described in Section 7.1, the following initial and boundary conditions were given:

$$u(x,0) = \begin{cases} \dfrac{2(4 - x)}{3} & 1 \leq x \leq 2.5 \\ 1 & x > 2.5 \end{cases} \tag{B.8}$$

$$u(L_1,t) = 2 \tag{B.9}$$

In the equations above x varies between L_1=1 m and L_2=11 m, and the constant in Eq. (b.1) is c = 1 m/sec.

The solution for this problem can be obtained along each characteristic curve given by Eq. (B.6) and a chosen x-coordinate ξ. Let ξ=1, for example. Then, the characteristic curve associated to this point is

$$x = 1 + t \tag{B.10}$$

and the solution given by Eq. (B.8) at time t=0 and x=1 is

$$u(0,1) = 2 \tag{B.11}$$

According to Eq. (B.7) this solution is valid along the characteristic curve given by Eq. (B.10). Thus, for t_1=1.5 sec, t_2= 3.0 sec, t_3=4.5 sec, t_4=6.0 sec, t_5=7.5 sec and t_6=9.0 sec, Eq. (B.10) gives the correspondent coordinates where the solution is the same u=2. These coordinates are x_1=2.5 m, x_2=4.0 m, x_3=5.5 m, x_4=7.0 m, x_5=8.5 m and x_6=10 m. The same procedure is repeated to determine the solution along any other characteristic curve. The results are shown in Fig. 7.1 and represent the exact solution for the linearized inviscid Burgers' equation.

APPENDIX C

THE SAAGG METHOD AND COMPUTATIONAL EFFICIENCY

The SAAGG method can greatly reduce the number of grid points needed to obtain solutions of partial differential equations by using finite-difference methods (FDMs). This reduction in the number of grid points enhances the computational efficiency of the FDMs when the SAAGG method is used for generating non-equally grid points in the physical domain.

Computational efficiency can be estimated by quantifying the amount of computer memory used and the processing time (i.e. CPU and I/O time) needed to complete a computing task. In this appendix, the improvements in computational efficiency gained by using the SAAGG method are described.

One-dimensional Problems

In this section, the processing time and the computer memory used to obtain two solutions of the one-dimensional problem presented in Section 6.3 are compared. One solution was obtained by using a FDM with 21 non-equally spaced grid points distributed in the physical domain by the SAAGG method. The other solution was obtained by using the same FDM with 251 equally spaced grid points distributed in the physical domain so that the minimum grid spacing was equal to the minimum grid spacing obtained by using the SAAGG method.

The exact solution of that problem and the solutions obtained by using non-equally spaced and equally spaced grid points are shown in Fig. C.1. This figure shows that in order to obtain solutions with approximately the same accuracy it was necessary to use 251 equally spaced grid points instead of the 21 non-equally spaced grid points generated by using the SAAGG method. This means that the number of grid points was increased by a factor of 12 and consequently the computational efficiency was decreased. The processing time and the computer memory needed to obtain these two solutions are compared in Table C.1. From this table it can be seen that by using the SAAGG method instead of using equally spaced grid points the processing time and the computer memory can be reduced by a factor of 2.33 and 4.28, respectively. Here, it is noted that the problem described in Section 6.3 is very simple and does not require much computational efforts to be solved. Hence, the ratio between the increasing in the processing time per grid point for using the SAAGG method (0.56-0.11) and the processing time per grid point for solving the problem without the SAAGG method (0.11) is approximately equal to 4.0. This ratio is reduced by orders of magnitude when the problem to be solved is more complex because in this case the processing time expended just for using the SAAGG method will be a small fraction of the total processing time.

Multidimensional Problems

For multidimensional problems, two cases can be considered. The first is when the solutions are obtained by using explicit FDMs and the second is when the solutions are obtained by using implicit FDM.

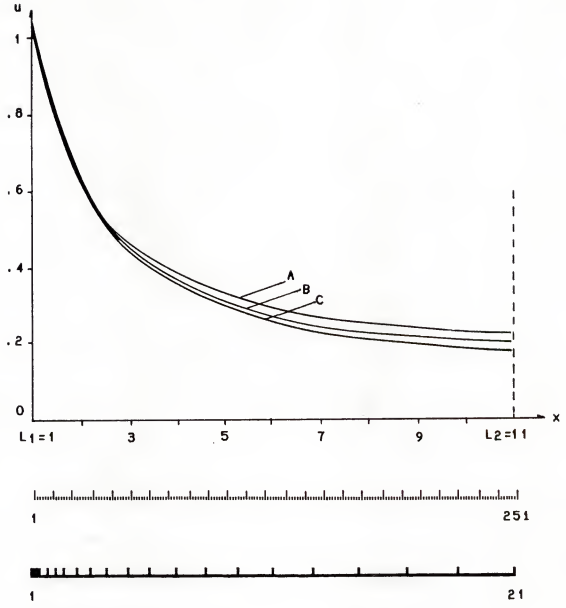


Figure C.1 Solution obtained for the initial-value problem given by Eqs. (6.3.1) and (6.3.2). The units for x and u are m and m/sec, respectively.

(A) Numerical solution obtained with 251 equally spaced grid points.

(B) Numerical solution obtained with 21 non-equally spaced grid points generated by using the SAAGG method.

(C) Exact solution

Table C.1* Comparison of Computational Efficiency

	SAAGG METHOD	EQUALLY SPACED METHOD
Number of Grid Points	21	251
Normalized Minimum Grid Points $\Delta x_{min}/(L_2-L_1)$	0.004	0.004
Processing Time T_t (sec)	11.75	27.46
Processing Time Per Grid Point T_p (sec/point)	0.56	0.11
Memory (Byte)	105x8	502x8

* All computing done on a Zenith PC-ZW-158-43

Explicit Methods

When using explicit FDMs to obtain solutions for multidimensional problems, the processing time is approximately equal to the processing time per grid obtained for one-dimensional problems multiplied by the number of dimensions (i.e. 2 for 2-D and 3 for 3-D) and by the total number of grid points. Thus, the processing time for a multidimensional problem can be expressed by

$$T_t = D\ T_p\ (IL \times JL) \qquad (C.1)$$

where T_t and T_p are defined in table C.1, D is the number of dimensions and IL and JL are the number of points in each grid line in the x- and y-direction, respectively.

From the above equation it can be seen that if the number of grid points in x- and y-direction were multiplied by a and b, respectively, the time processing would be multiplied by a factor equal to a times b.

Taking for example the same figures shown in Table C.1 the processing time for a two-dimensional problem would be 8 min and 14 sec by using the SAAGG method and 3 h and 51 min by using equally spaced grid points. For a three-dimensional problem, the time processing would be 12 min and 21 sec by using the SAAGG method and 5 h and 21 min by using equally spaced grid points.

Implicit Methods

In order to solve a two-dimensional problem by using implicit FDMs, the following matrix equation needed to be solved:

$$[A]x = b \tag{C.2}$$

where the matrix [A] is of order N and N=ILxJL.

The number of arithmetical operations necessary for solving Eq.(C.2) by using Gauss elimination is of order of $O(N^3)$. Thus if the number of grid points in x- and y-direction were multiplied by a constant a and b, respectively, the number of arithmetical operations would be multiplied by a factor $(ab)^3$. In this case the SAAGG method is tremendously advantageous because by reducing the number of grid points in each coordinate direction by order of 10, for example, the number of arithmetical operations will be reduced by order of 10^6.

REFERENCES

1. Thompson, Joe F., "Grid Generation Techniques in Computational Fluid Dynamics," AIAA Journal, Vol. 22, No. 11, pp. 1505-1523, 1984.

2. Dale, A. Anderson, "Adaptive Grid Methods for Partial Differential Equations," Advances in Grid Generation, ASME, Vol. 5, pp. 1-117, 1983.

3. Shih, T. I-P., Finite-Difference Methods in Computational Fluid Dynamics, Prentice-Hall, Englewood Cliffs, N.J., to be published.

4. Mastin, C. W., "Error Induced By Coordinate Systems," Numerical Grid Generation, Editor: J.F. Thompson, North-Holland, New York, p. 31, 1981.

5. Thompson, J.F., Zahir, U. A. W. and Mastin, C. W., "Boundary-Fitted Coordinate Systems for Numerical Solution of Partial Differential Equations — A Review," J. Comp. Phys., Vol. 47, pp. 1-108, 1982.

6. Hindman, R.G. and Spencer, J. "A New Approach to Truly Adaptive Grid Generation," AIAA Paper 83-0450, AIAA 21st Aerospace Sciences Meeting, Reno, January 1983.

7. Saltzman, J. and Brackbill, J.U., "Application and Generalization of Variational Methods for Generating Adaptive Meshes," Numerical Grid Generation, Editor: J.F. Thompson, North-Holland, New York, pp. 865-884, 1982.

8. Brackbill, J.U. and Saltzman, J., "Adaptive Zoning for Singular Problems in Two Dimensions," J. Comp. Phys., Vol. 46, pp. 342-268, 1982.

9. Saltzman, J., "A Variational Method for Generating Multidimensional Adaptive Grids," Courant Math. and Comp. Lab. DOE/ER/3077-174, 1982.

10. Saltzman, J., "Variational Methods for Generating Meshes on Surfaces in Three Dimensions," J. Comp. Phys., Vol. 63, pp. 1-19, 1986.

11. Winslow, A. M., "Equipotential Zoning of Two-Dimensional Meshes," J.Comp. Phys., Vol. 149, pp. 152-172, 1966.

12. Bell, J. B. and Shubin, G. R., "An Adaptive Grid Finite Difference Method for Conservation Laws,"J. Comp. Phys., Vol. 52, pp. 569-591 1983.

13. Rai, M. M. and Anderson, D. A., "Grid Evolution in Time Asymptotic problems," J. Comp. Phys., Vol. 43, pp. 327-344, 1981.

14. Rai, M. M. and Anderson, D .A., "The Use of Adaptive Grid in Conjunction with Shock Capturing Methods," AIAA Paper 81-1012, pp. 156-165, AIAA 5th Computational Fluid Dynamics Conference, Palo Alto, California, June 1981.

15. Dwyer, H.A., Kee, R.J. and Sanders B.R., "An Adaptive Grid Method for Problems in Fluid Mechanics and Heat Transfer," AIAA Journal, Vol. 18, pp. 1205-1212, 1980.

16. Gnoffo, P.A., "A Vectorized, Finite Volume, Adaptive-Grid Algorithm for Navier-Stokes Calculations," Numerical Grid Generation, Editor: J.F. Thompson, North-Holland, New York, pp. 819-835, 1982.

17. Gnoffo, P. A., "A Finite Volume, Adaptive-Grid Algorithm Applied to Planetary Entry Problems," AIAA Journal, Vol. 21, pp. 1249-1254, 1983.

18. Nakahashi, K. and Deiwert, G.S., "Self-Adaptive Grid Method With Applications to Airfoil Flow," AIAA Journal, Vol. 25, pp. 513-520, 1987.

19. Nakahashi, K. and Deiwert, G.S., "Three-dimensional Adaptive Grid Method," AIAA Journal, Vol. 24, pp. 948-954, 1986.

20. Smooke, M.D. and Koszykowski, M.L.,"Fully Adaptive Solutions of One-dimensional Mixed Initial-boundary Value Problems with Applications to Unstable Problems in Combustion," SIAM J. SCI. STAT. Comp., Vol. 7, No. 1, pp. 301-321, 1986.

21. Reuven, Y., Smooke, M.D. and Rabitz, H., "Sensitivity Analyzes of Boundary Value Problems: Application to Nonlinear Reaction-Diffusion Systems," J. Comp. Phys., Vol. 64, pp. 27-55, 1986.

22. Smooke, M.D., "Use of Adaptive Methods in Premixed Combustion," AIChE Journal, Vol. 32, No. 8, pp. 1233-1241, 1986.

23. Anderson, D.A., Tannehill, J.C. and Pletcher, R.H., Computational Fluid Mechanics and Heat transfer, McGraw-Hill, New York, 1984.

24. Vinokur M., "On One-dimensional Stretching Functions for Finite-Difference Calculations," J. Comp. Phys., Vol. 50, pp. 215-591, 1983.

25. Thompson, J.F., Warsi, Z.U.A. and Mastin, C.W., Numerical Grid Generation: Foundations and Applications", North-Holland, New York, 1985.

26. Smith, R.E., "Two-Boundary Grid Generation for the Solution of the Three-dimensional Navier-Stokes Equations," NASA TM-83123, NASA, Langley Research Center, Hampton, Virginia, 1981.

27. Smith, R.E., "Algebraic Grid Generation," Numerical Grid Generation, Editor: J.F. Thompson, North-Holland, New York, pp. 137-170, 1982.

28. Yang, S.-L. and Shih, T.I-P., "An Algebraic Grid Generation Technique for Time-Varying Two-Dimensional Spatial Domains," Int. Journal for Num. Meth. in Fluid, Vol. 6, pp. 291-304, 1986.

29. Roache, P.J., Computational Fluid Dynamics, Hermosa Publishers, Albuquerque, New Mexico, pp. 248-253, 1985.

30. Fletcher, C.A.J., "Burgers' Equation: A Model for All Reasons," Numerical Solutions of Partial Differential Equations, Editor: J. Noye, North-Holland, Amsterdan, pp. 139-225, 1982.

31. Mickens, R. E., "Exact Solutions to Difference Equations Models of Burgers' Equation," Numerical Methods for Partial Differential Equations, Vol. 2, pp. 123-129, 1986.

BIOGRAPHICAL SKETCH

Julio S. Dolce da Silva was born on March, 13, 1945, in Brazil. He is married to Marianna L.B. Dolce da Silva and he is the father of two children. One is Clarissa, presently a freshman at the University of Florida and the other is George, a sophomore at the Gainesville High School.

In 1965, he received a bachelor's degree in ordnance from the "Academia Militar das Agulhas Negras" (The Brazilian Army Academy). One year later, he was commissioned as a lieutenant in the Brazilian Army. At present, he holds the rank of lieutenant-colonel.

In 1970, Mr. Dolce da Silva was selected to attend the "Instituto Militar de Engenharia." He was graduated from this institute in 1973 with the degree of "Engenheiro Mecânico e Automóvel" (bachelor's degree in mechanical engineering). For his performance in undergraduate studies, Mr. Dolce da Silva received the "Henry Ford II Award."

In 1981, he was selected to enter upon graduate study at the "Instituto Militar de Engenharia." In 1982, he was awarded the degree of "Mestre em Ciências" (Master of Sciences). The title of his master thesis was "Charts of Combustion for Multifuel Engines." After his graduation, he was invited to join the "Centro Tecnológico do Exército" as a research engineer and in 1983, he was appointed as an instructor to teach the course of Internal Combustion Engines at the "Instituto Militar de Engenharia."

In 1984, Mr. Dolce da Silva received the Medal of Honor from the Brazilian Army for twenty years of distinguished service as an Army officer involved with engineering research of national and international significance.

In 1985, Mr. Dolce da Silva was selected by the Brazilian Army to be enrolled in the Graduate School of the University of Florida. Since then, he has been engaged in studies and research to fulfill the University of Florida requirements for the degree of Doctor of Philosophy.

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

Dr. V.P. Roan, Chairman
Professor of Mechanical
Engineering

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

Dr. T.I-P. Shih, Cochairman
Associate Professor of
Mechanical Engineering

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

Dr. R.B. Gaither,
Professor of Mechanical
Engineering

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

Dr. H.A. Ingley,
Associate Professor of
Mechanical Engineering

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

J. Hammack

Dr. J. Hammack, Jr.,
Professor of Engineering Sciences

This dissertation was submitted to the Graduate Faculty of the College of Engineering and to the Graduate School and was accepted as partial fulfillment of the requirements for the degree of Doctor of Philosophy.

April 1988

Hubert A. Bruns

Dean, College of Egineering

Dean, Graduate School